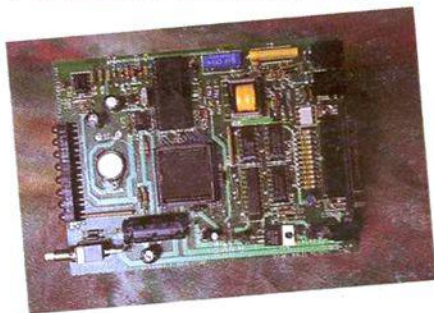
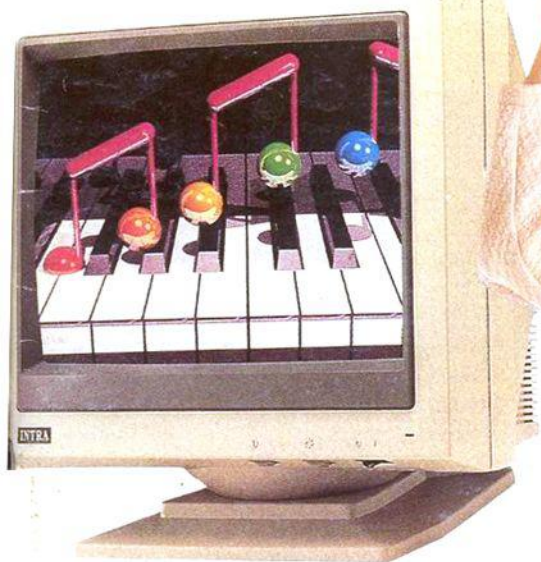
Anno XI - N. 84 - 25 Maggio 1991 - Sped. Abb. Post. Gr III/70 - CR - Distr.: Parrini

TELEMATICA
I modem ultraveloci

POSTSCRIPT
Un linguaggio risparmia - byte

MUSICA 64
Risposta alla sfida

DIGITALIZZATORI
Video, video che passione!

DTP, AMIGA IMPAGINA

La vostra posta • Word 5.5 • Turbo C - Assembly 80X86 • Gioco del 15 • T. Pascal • Amos Basic Amiga • Amiga Audiomaster • Files Batch • Una clessidra per Amiga • La gestione delle stringhe in "C" • La Rad di Amiga • Come scambiar file tra Amiga e Ms-Dos

Lire 6.000

Personal COMPUTER

La rivista per utenti di personal computer e workstation

Esclusivo: Gestione C/C dBASE III Plus

IN EDICOLA

INCHIESTA

Conviene comprare un computer usato?

SPECIALE

Telefonia cellulare

IN PROVA

Norton Editor 2.0

Harvard Graphics 2.3 in italiano

Viewlink

Textor 5

Epson Lq 2550

Epson Ax3/25

Hyundai 386N Plus

Unibit Tsx/3000

Anteprima QuattroPro 3.0

Sommario

Spazio 64

Amiga

Amiga + Ms - Dos

Mondo Dos

COMMODORE COMPUTER CLUB

Direttore: Alessandro de Simone
Coordinatore: Marco Miotti

Redazione / Collaboratori:
Davide Ardizzone - Claudio Baiocchi
Luigi Callegari - Umbero Colapicchioni
Donato De Luca - Carlo D'Ippolito
Valerio Ferri - Michele Maggi
Giancarlo Mariani - Domenico Pavone
Armando Sforzi - Dario Pistella
Fabio Sorgato- Valentino Spataro
Franco Rodella- Stefano Simonelli
Luca Viola

Direzione:
Via Mosè, 22 cap. 20090 OPERA (Mi)

Telefono 02 / 55.50.03.10
Fax 02 / 57.60.30.39
BBS 02 / 52.49.211

Pubblicità:
Leandro Nencioni (dir. vendite)
Via Mose', 22 20090 Opera (Mi)
tel. 02 / 55.50.03.10

Emilia Romagna:
Spazio E
P.zza Roosvelt, 4 cap. 40123 Bologna
Tel. 051 / 23.69.79

Toscana, Marche, Umbria
Mercurio s.r.l. Via Rodari, 9
S. G.nni Valdarno (Ar)
Tel. 055 / 94.74.44

Lazio, Campania
Spazio Nuovo
Via P. Foscari, 70
cap. 00139 Roma
tel. 06 / 81.09.679

Abbonamenti: Liliana Spina
Arretrati e s/w: Lucia Dominoni

Tariffe: Prezzo per copia L. 6000
Abbonamento annuo (11 fascicoli) L. 60000
Estero: L. 100000 - Indirizzare versamenti a:
Systems Editoriale Srl c/c 37952207 oppure
inviare comune assegno bancario non
trasferibile e barrato due volte a:
Systems Editoriale Srl (servizio arretrati)
Via Mose', 22
cap. 20090 OPERA (Mi)

Composizione: Systems Editoriale
La Litografica Srl Busto Arsizio (Va)

Registrazioni: Tribunale di Milano
n. 370 del 2/10/82

Direttore Responsabile: Michele Di Pisa

Spedizioni in abbonamento postale gruppo
III. Pubblicita' inferiore al 70%

Distributore: Parrini - Milano

Periodici Systems: Commodore Club (disco)
- Commodore Computer Club - Commodore
Computer Club (disco, produzione tedesca) -
Computer - Computer disco - Hospital
Management - Jonathan - Nursing '90 - PC
Programm (disco) - Personal Computer -
Security - Software Club (cassetta ed.
italiana) - Videoteca VR Videoregistrare

Dalla protezione al Virus

Nel 1986 il Chaos Computer Club organizzò un congresso che aveva, come scopo principale, la discussione su un fenomeno emergente: i virus informatici.

Da quella data i *mass media* si occuparono dei virus, spesso a sproposito (dimostrando una totale disinformazione di base) e molte furono le notizie al riguardo, soprattutto per quanto concerneva la "spettacolarità" delle epidemie.

Di certo, da quella data il numero dei virus conosciuti aumentò in maniera notevole; i danni che ne derivarono, poi, crearono serie difficoltà sia per gli utenti che per i produttori.

Molte, infatti, furono le Software House che, per limitare la diffusione di copie pirata dei propri prodotti, inserirono (nei package commercializzati) ogni sorta di virus, dai Cavalli di Troia alle bombe a tempo, ai virus distruttivi a quelli che generavano malfunzionamenti di vario tipo. Purtroppo, e fortunatamente solo in casi particolari, tali espedienti si attivavano anche se veniva utilizzata la copia originale del programma. Inutile soffermarsi sulle conseguenze.

Molti creatori di virus, d'altra parte, riuscirono, nelle proprie sperimentazioni, ad infettare in modo permanente il proprio elaboratore dimostrando, loro malgrado, di essere riusciti a realizzare un virus a prova di antibiotico. Anche in questo caso, meglio non soffermarsi su commenti più che ovvi.

Sul terzo fronte, infine, l'alto prezzo al pubblico di programmi professionali non protetti (certamente sofisticati ed utili) non contribuivano di certo a limitare il fenomeno della pirateria, nel cui circuito si inserivano, appunto, disinvolti *monatti* che si divertivano a divulgare copie infettate di detti programmi.

Chiariamo: la pirateria del software, piaccia o no, ha contribuito in maniera massiccia alla diffusione dell'informatica di massa; la notevole sofisticazione dell'hardware e del software, d'altra parte, ha contribuito allo sviluppo di tecniche di protezioni (**sempre** inefficaci) e di epidemie (**spesso** "efficaci", nel senso deteriore del termine).

A pagarne le conseguenze rischiamo di essere noi tutti, dagli smanettoni incalliti (chi ha sempre e comunque verificato la presenza di virus su ciascun dischetto che entrava nel proprio drive, scagli la prima pietra) agli utenti inesperti (chi non ha mai inserito un giochino nel computer dell'azienda in cui lavora, scagli la seconda pietra).

Basta un attimo di distrazione e un virus di nuova concezione, o un Cavallo di Troia particolarmente cattivo, ed ecco che un hard disk viene formattato, una Ram portata al limite di sopportazione, il braccio di un drive scagliato contro il fine corsa.

La smilitarizzazione è iniziata, da un po' di tempo, rendendosi conto che il numero di megatoni accumulato per ciascun essere umano era eccessivo rispetto ai "vantaggi" che potevano derivare da un eventuale uso dell'arma nucleare.

Nel campo dell'informatica penso che stiamo raggiungendo lo stesso livello di incoscienza. Occorre fermarsi, e subito.

Nei prossimi numeri vedremo di approfondire l'argomento, confortati, me lo auguro, dal commento dei nostri lettori che sono invitati, fin da ora, a comunicare le proprie idee a proposito.

Alessandro de Simone

LA VOSTRA POSTA

(a cura di A. de Simone)

Chi non salta è...

Vorrei sapere come intervenire, in un listato Basic, affinchè un salto (prodotto, ad esempio, da GoTo 50) esegua la sola riga 50 e non quelle successive; il tutto, ovviamente, adoperando anche il linguaggio macchina.
(Giuseppe)

Penso che il nostro lettore intenda escludere l'uso delle subroutine (**GoSub... Return**), che risolverebbero il problema in modo più che efficiente (e, soprattutto, semplice).

Una modifica del genere, infatti, richiederebbe l'intervento sulla parte in linguaggio macchina che gestisce il riconoscimento dei comandi Basic. Si dovrebbe, inoltre, far capire all'interprete in quali casi il comando GoTo deve essere interpretato in modo tradizionale e quando, al contrario, bisogna eseguire la sola riga interessata dalla nuova implementazione. Il tutto per risparmiare la duplicazione delle righe Basic per trasformarle in tranquillissime subroutine; ne vale la pena?

Gestionali per Amiga

Perchè non si riescono a trovare in commercio programmi gestionali per Amiga?
(Alberto Pretto)
(Sergio Carta)

Gli acquirenti di Amiga posseggono, nel 99.99% dei casi, il modello Amiga 500; di questi, il 99.999% non ha ritenuto utile acquistare anche il disco rigido, periferica rigorosamente indispensabile per lavorare in modo *dignitoso* con programmi professionali quali, appunto, i package gestionali.

Delle 35 persone residue (che posseggono il disco rigido), solo 4 / 5 utilizzerebbero l'Amiga in modo professionale. Di queste, le 2 che hanno necessità gestionali le risolvono adoperando uno dei 3 computer **Ms - Dos** che certamente possiedono (oltre all'Amiga).

Domanda: quale Software House, oggi, rischierebbe di creare software di un certo tipo per tenerlo, invenduto, in magazzino?

Il secondo lettore afferma di conoscere *numerosi* studi professionali che utilizzano l'Amiga. Bene, vorrei l'indirizzo ed il telefono di questi ultimi, allo scopo di informarmi sull'intera dotazione di apparecchiature informatiche presenti nei suddetti studi.

Poi, semmai, ne riparleremo.

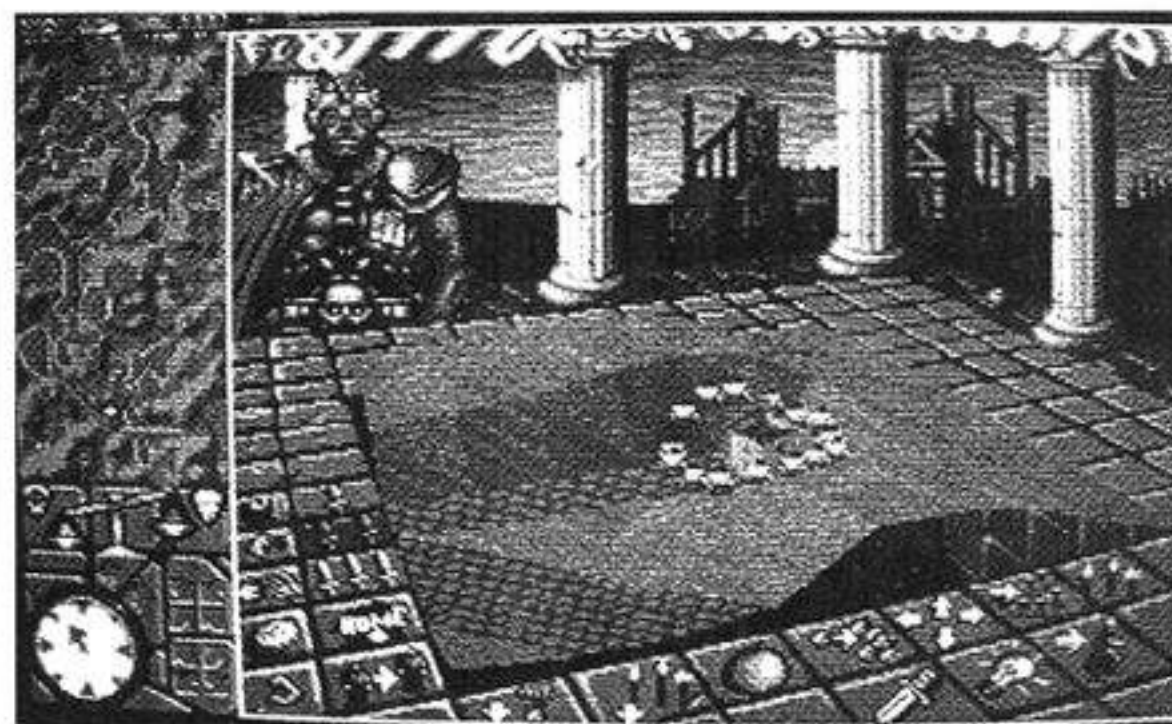
Collaborazioni

Le proposte di collaborazione si accettano esclusivamente per telefono, allo scopo di valutare in tempi brevissimi la reale rispondenza dei vostri lavori alle nostre necessità. Qualunque altra forma di "contatto" (fax, lettere, BBS ed altro) viene rigorosamente ignorata.

Quale Ms- Dos

A giugno comprerò un Ms - Dos compatibile, ma sono indeciso sulle caratteristiche che deve possedere. Potete darmi qualche consiglio?
(Gianluca Caminiti - R. Cal.)

No, ovviamente, dal momento che nella lettera non viene specificato l'uso che intendi farne.

Tuttavia, nella mia infinita bontà, cercherò di dare egualmente qualche consigliuccio. Se non vi sono problemi economici, è meglio un **80386** (*non* SX) che un 80286; se, invece, vi sono problemi eco-

La sfida del mese

Stavolta lo spazio, sempre più tiranno,ci relega in questo angusto angolino.

Poco male, anche perchè il lavoro che vi invitiamo a svolgere non è molto complesso da descrivere.

Come certamente saprete sapete, sia con sistemi **Amiga** che con sistemi **Ms - Dos** è possibile usare il comando **Copy** per spostare da una directory all'altra (e/o da un disco all'altro) un qualunque file (o gruppi di files) servendosi, magari, dei caratteri **WildCards** (asterisco e punto di domanda per Ms - Dos; cancelletto e punto di domanda per Amiga).

Ciò, purtroppo, non è valido per le **directory intere**, contenenti, cioè, altre subdirectory oltre ad eventuali files, tranne che per Amiga che, sfruttando il comando **Rename**, consente tali spostamenti (ma solo se si verificano sul medesimo disco).

Programmi di pubblico dominio (**DirCopy** e **DirErase** per **Ms - Dos**, ad esempio) svolgono egualmente la comoda funzione di copiare intere directory, e loro contenuto, da un disco all'altro.

Ed ecco la sfida: utilizzando tali programmi di utilità, creare un file Batch (oppure un vero e proprio programma scritto in Pascal, Basic, C, Assembler) in grado di svolgere la funzione Muovi, allo scopo di spostare una directory da un disco ad un altro effettuando, nel contempo, una verifica sulla reale possibilità della procedura.

Ad esempio, impartendo su un computer Ms - Dos un comando del tipo...

```
Muovi a:MiaDir c:
```

...la directory **Miadir** (e l'*intero* suo contenuto di files ed eventuali subdirectory) memorizzata sul disco **A:** sia trasferita sul disco **C:** cioè dapprima **copiata** e poi **cancellata** dalla posizione originale.

Naturalmente, **prima** di cancellarla, è assolutamente indispensabile verificare che il trasferimento sia stato realmente effettuato senza errori. Questi, come è noto, potrebbero verificarsi in vari casi:

- Inesistenza del drive (e/o subdir) di destinazione richiesto.
- Tacchetta di protezione attiva.
- Disk full.
- Nome di subdir già presente sul dischetto destinazione.
- Disco di destinazione non formattato o contenente errori di vario tipo.

In tutti i casi segnalati, quindi, il batch file (o procedura che dir si voglia) deve **interrompere** la procedura di trasferimento, **emettere** opportuna segnalazione di errore e, soprattutto, **astenersi** dal cancellare la directory dal disco sorgente!

Come al solito, possono partecipare alla sfida tutti coloro che lavorano abitualmente con computer **Amiga** oppure **Ms - Dos**; è buona norma, in ogni caso, telefonare in Redazione **prima** dell'invio del materiale.

Verranno privilegiati, per la pubblicazione, le procedure **più brevi da digitare** e, ovviamente, solo quelle che realmente risponderanno in modo completo alla sfida lanciata.

nomici (ma non tanto da esser costretti nell'ambito dell'8086, ormai un po' obsoleto e lento) meglio un **80286** che un 80386-Sx. Disco rigido **indispensabile** (lascia perdere il 20 mega, deve essere da almeno **40** mega, ideale se 80 o superiore, ma senza esagerare): meglio rinunciare all'informatica **seria** che possedere un computer privo di hard disk (che, comunque, può essere acquistato in un secondo momento). Memoria minima 1 mega (meglio se **2** mega; con l'80386 meglio ancora 4 mega); porte: 2 **Rs-232** (la prima per il mouse, la seconda per il futuro modem o altri accessori basati su Rs-232); una parallela (Centronics, insomma, per la stampante); il numero minimo dei connettori interni dipende dalle schede che intendi inserire, ora ed in futuro. Tener presente che un connettore viene occupato dalla scheda **grafica**, un altro dal **controller** (di solito valido per due hard disk e due floppy; alcuni controller incorporano anche la presa per il joy e, addirittura, una porta seriale); meglio prevedere un modem interno (e quindi il connettore corrispondente) ed un altro accessorio di cui, domani, potrà venire la voglia (scheda sonora, scanner); controllare se c'è lo spazio per tutti i drive che si intendono montare, ora ed in futuro.

P.S: spero che tu abbia già provveduto a vendere il tuo vecchio computer (quale? chissà...) al compagno di scuola di cui parli nella lettera. Tra breve, infatti, il suo valore (del computer, non del compagno, che vale meno) tenderà ad annullarsi dal momento che nessuno si occuperà di scrivere ancora software specifico. Credimi, hai fatto bene a venderlo!

Libri di Informatica

Una notizia ghiotta per gli utenti di **Amiga** ed **Ms - Dos**. La "Libreria dell'Informatica" (**Galleria Pattari, 2 c.a.p. 20122 Milano - Tel./Fax 02/8690375**) risulta particolarmente fornita di testi e software di provenienza U.S.A. (oltre che, ovviamente, di edizioni italiane). Tra gli scaffali abbiamo notato la presenza dei mitici manuali Amiga **Rom Kernel** e perfino il package per l'installazione di un mini **Unix** su Amiga ed Ms - Dos. Ma la notizia più bella è un'altra: la Libreria dell'Informatica effettua **vendite per corrispondenza in tutt'Italia**. Telefonate per ricevere il catalogo (costantemente aggiornato) e dite che il numero di telefono ve lo abbiamo dato noi!

di Alessandro de Simone

Microsoft Word 5.5, dal word processor al DTP

La presentazione del nuovo prodotto Microsoft pone alcuni interrogativi sul progresso compiuto dalle software house sui programmi di editoria; ne approfittiamo per chiarire alcuni concetti

Un articolo che tratti dei progressi compiuti in un qualsiasi campo, dovrebbe iniziare un discorso *ab ovo* allo scopo di meglio chiarire da dove si è partiti e, soprattutto, dove si è arrivati.

Se, poi, si cerca di chiarire anche il perché, le parole non risulterebbero sprecate invano.

Non staremo, però, a ripeterci su argomenti già triti e ritriti (vedi, a tal proposito, la chiacchierata su **come usare un word processor**, pubblicata sul numero scorso); ci basterà effettuare alcune considerazioni che, trascurate a torto da una gran quantità di utenti, mostrano la loro importanza soprattutto esaminando la modesta qualità dei lavori svolti dagli utenti stessi.

Maledetta macchina

Il mercato mondiale delle macchine da scrivere, per fortuna nostra (e del genere umano in generale), va rapidamente affievolendosi fino a scomparire del tutto entro qualche anno.

La felicità (in parte sadica, lo ammetto) che trapela a malapena dalle parole precedenti, ha la sua ragion d'essere nella constatazione della rovina prodotta, da tale macchina infernale, nelle menti delle giovani generazioni.

Conosco personalmente numerosi e validi programmatori che sono in grado di far resuscitare files ormai persi per sempre, scrivere listati per inserirsi subdolamente in banche dati segrete, realizzare programmi di strabilianti effetti sonori e grafici. Tutti questi personaggi, però, falliscono miseramente se sottoposti alla prova della stesura di un semplice documento, tipo *"Spettabile Ditta... Distinti saluti"*.

La dettatura di una ventina di righe, realizzata usando un banale word processor, mette in luce i devastanti effetti prodotti dal virus *"scripturae machina"* dopo anni di utilizzo della dannata macchina.

La *forma mentis* dei giovani (e non) di cui sopra sembra appartenere ai tempi pre-moderni della rivoluzione industriale: molti, infatti, giunti alla fine del rigo video premono il tasto Return (ignorando che un qualsiasi w/p "manda a capo" in modo automatico); alcuni, per evidenziare una frase, la scrivono tutta in maiuscolo; altri, sciagurati, scrivono tabelle usando i punti esclamativi per delimitarne i campi ed altri ancora, *incredibile visu dictuque*, realizzano "rientranze" di testo impiegando, riga per riga, la pressione della barra spaziatrice. Chi, poi, cerca di cen-

GG&G FORUM
News from the Publishing Division

trare un titolo "a video" (servendosi, ancora, della barra spaziatrice) non può nemmeno esser fucilato sul posto, a causa di una mal interpretata legge sulla libertà di stampa.

Tutti gli effetti finora descritti sono addebitabili ad anni di uso inconsulto della macchina da scrivere.

Dal momento che (ne siamo sicuri) molti di coloro che ci leggono commettono almeno una ventina tra gli errori descritti (e, ovviamente, non hanno il minimo sospetto che siano "errori") è opportuno fare una pausa e spiegare che un computer è, per fortuna (nostra e del genere umano) alquanto diverso da una volgare, obsoleta ed oscena macchina da scrivere.

C'è macchina e macchina

Purtroppo le primitive attrezzature per la stesura meccanizzata di documenti presentavano limiti, accettati fino a pochi anni orsono.

Chi scriveva un documento, infatti, sapeva benissimo che doveva controllare costantemente il rigo che a mano a mano veniva scritto, in modo da valutare, pur se con qualche approssimazione, il momento in cui decidere l'eventuale troncatura in sillabe di una parola.

A tale scopo, anzi, i fabbricanti di macchine da scrivere avevano posizionato un campanello che, quando il margine destro si approssimava alla fine del rigo, veniva percosso dal rullo, in modo da avvertire anche l'utente più distratto. Gli uffici di una volta, quindi, a parte il rumore (tasto per tasto) prodotto dai martelletti, erano allietati dal suono dei campanacci tipo monatti-in-cerca-di-appestati-morti (scusate, mi sono lasciato trascinare dall'entusiasmo).

Ancora oggi, durante le interviste a famosi giornalisti e degni romanzieri, questi ignobili personaggi si fanno fotografare con pipa in bocca e (ben posta in primo piano, dinanzi a loro) una macchina da scrivere (nemmeno elettrica), a dimostrazione della totale ignoranza nella quale nuotano e dalla quale traggono spunto per i loro scritti.

Polemiche a parte, non dobbiamo dimenticare che il progresso tecnologico ha messo a disposizione il computer il cui scopo principale è quello di evitare errori, fatiche e preoccupazioni. Altrimenti, che ci sta a fare?

Libertà di stampa

La macchina da scrivere era uno strumento che permetteva due **vantaggi principali**: il primo consisteva nella possibilità di stesura di documenti estremamente leggibili, indipendenti dalla qualità della calligrafia di chi scriveva a mano; il secondo, molto più importante, consentiva una velocità di scrittura incomparabilmente superiore a quella manuale, paragonabile solo a quella raggiungibile dalla stenografia.

Ancòra oggi, pochi si rendono conto (soprattutto perché nessuno lo dice apertamente) che anche chi utilizza solo due dita può raggiungere, dopo poche ore di esercizio, una velocità di scrittura almeno pari a quella normalmente consentita con carta e penna.

Se, poi, ci si dedica con una certa costanza (mezz'ora al giorno, non di più), la velocità di scrittura raggiungibile è talmente elevata che difficilmente, in seguito, si può fare a meno del sistema meccanizzato.

Dunque: **maggiore velocità, migliore qualità**. Che cosa si può pretendere di più? Molto ancòra, come vedremo subito.

Un problema irrisolvibile con le attrezzature obsolete (cioè con le macchine da scrivere) è costituito dagli errori di battitura, che si verificano, in media, almeno ogni dozzina di righe scritte (se si è fortunati). I **rimedi** (o meglio, i **palliativi**) offerti dalla precedente tecnologia, consistevano in foglietti cerati bianchi sui quali si ribatteva il carattere errato, per cancellarlo.

Se, però, c'era un'intera frase da cancellare (o, peggio, da aggiungere), l'unico rimedio consisteva nel ribattere il foglio; e se questo era il terzo di (ad esempio) cinque, la frase inserita costringeva a ribattere, spesso, anche i fogli numero quattro e cinque; e magari a creare un sesto foglio sul quale, mìsere, trovavano posto le ultime quattro parole del documento...

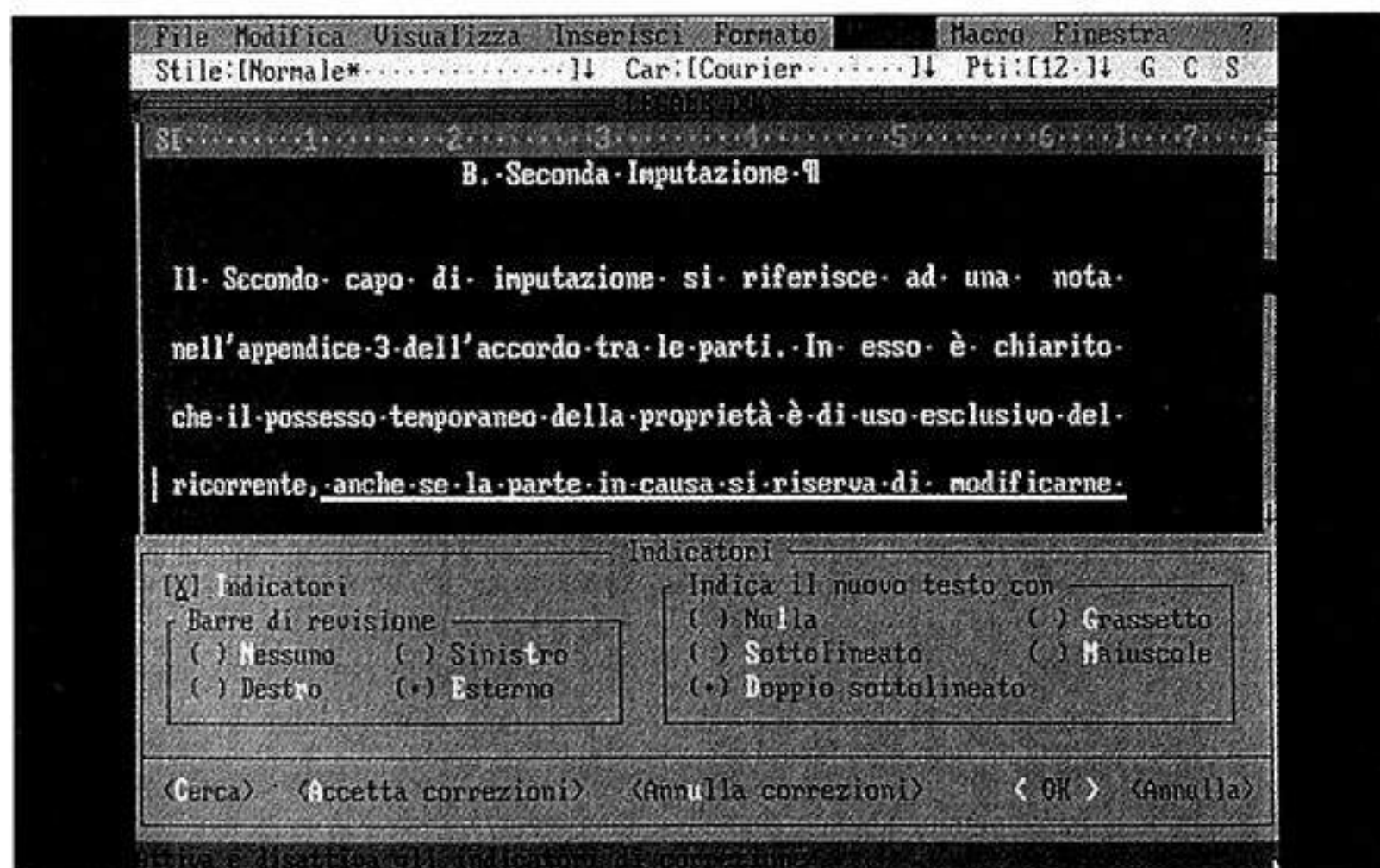
Tutta 'sta tiritera costringeva il personaggio dell'epoca (la famosa segretaria, che alla fine scappava in Brasile con il principale non sempre per amore, ma soprattutto per non sentir più parlare di macchine da scrivere) a prendere appunti del documento, batterlo una prima volta, valutare le correzioni per batterlo una seconda e (se tutto andava bene) definitiva volta.

Nonostante le difficoltà descritte, la macchina da scrivere consentiva una qualità ed una velocità di stesura non paragonabili al sistema manuale, l'unico alternativo per quei tempi.

Oggi il computer consente di scrivere liberamente un qualsiasi documento senza badare a:

- Centrature di titoletti
- Giustificazione del testo
- Impostazione di margini
- Rientranze e fuoruscite del testo
- Numerazione delle pagine

Molti altri sono i vantaggi offerti da un sistema di videoscrittura moderno (for-

mattazioni particolari e divisione in sillabe, tanto per citare le più semplici); se ci siamo limitati ad indicarne solo cinque, è perché un qualsiasi word processor (se non, addirittura, un Text Editor) è in grado di offrire tali opportunità.

In pratica, chi digita un documento al computer si può permettere il lusso di scrivere come gli pare: in seguito, con la massima calma, sarà sempre possibile:

- Inserire frasi
- Cancellare (o spostare) caratteri, parole, intere parti di testo
- Impostare margini diversi da quelli precedenti
- Centrare (o allineare in modo diverso) titoletti o parole in genere
- Modificare rientranze e fuoruscite
- Verificare il numero di pagine che verranno realmente stampate (onde evitare, agendo sul numero di righe per pagina, di ottenere pagine troppo "vuote").

Le cause, insomma, che costringevano alla ribattitura di parti anche corrette del documento (nel caso, per esempio, di impostazione di nuovi margini) vengono eliminate usando un computer.

Il tempo medio per ottenere un qualsiasi testo "finito" (considerando le fasi di bozza, prima... seconda... ultima correzione e relative prima... seconda... ultima stampa) si riduceva drasticamente tanto da far affermare che **un utente munito di computer è in grado di produrre quanto tre utenti muniti di una semplice macchina da scrivere**.

Qualcuno, subito, pensò bene di utilizzare il tempo risparmiato non soltanto per consentire la stesura di una maggior *quantità* di documenti (la mole di lavoro, infatti, potrebbe non esser elevata) quanto per migliorare la **qualità** degli stessi; nasceva, così, l'idea del Desk Top Publishing.

Comunicazione

Una bella foto vale più di mille parole (figuriamoci, poi, un filmato, magari a colori e in suono stereo).

Nel campo del lavoro la maggior parte delle informazioni viaggia prevalentemente su due canali: scritto e parlato.

Se, soprattutto per telefono, si riesce a comunicare con una certa facilità (ma, spesso, con una certa fatica, e sempre con due soli interlocutori alla volta), nel caso di una **comunicazione scritta** sorgono diversi problemi: un comunicato, infatti, deve essere di rapida comprensione, chiaro e facilmente leggibile. Tutte qualità, queste, che non sempre si riescono a raggiungere, soprattutto in assenza di grafici o di disegni.

I progressi consentiti dai primi computer, quindi, si limitavano a sfruttare tutte le qualità delle stampanti che, a quei tempi, iniziavano la loro diffusione su larga scala: impostazione del *corsivo*, del **neretto**, del sottolineato, del MAIUSCOLETTO. Numerosi altri stili, spesso del tutto inutili (ombreggiato, scavato, tre D ed altri di analoga amenità), ampliavano tuttavia la scelta di stampa invitando alla sperimentazione. Si assistette, in breve, alla stesura di documenti che, in una pagina, presentavano dodici stili in venti formati diversi, per un totale di una cinquantina di parole al massimo.

Se, però, venivano usate con discrezione, le potenzialità offerte dai word processor consentivano la **stesura di documenti di notevole efficacia** che permettevano, vivaddio, la possibilità di eliminare per sempre l'utilizzo di caratteri maiuscoli per evidenziare una parte del testo.

Ai tempi delle macchine da scrivere, infatti, vi erano soltanto due modi per mettere in evidenza alcune parole in un testo: la sottolineatura ed il ricorso ai caratteri maiuscoli.

Oggi, per fortuna (nostra e del genere umano in generale), la possibilità di usare il carattere corsivo ed il neretto (ma anche il maiuscoletto non è da disprezzare) permettono di evidenziare parole in modo più discreto e, soprattutto, più efficace.

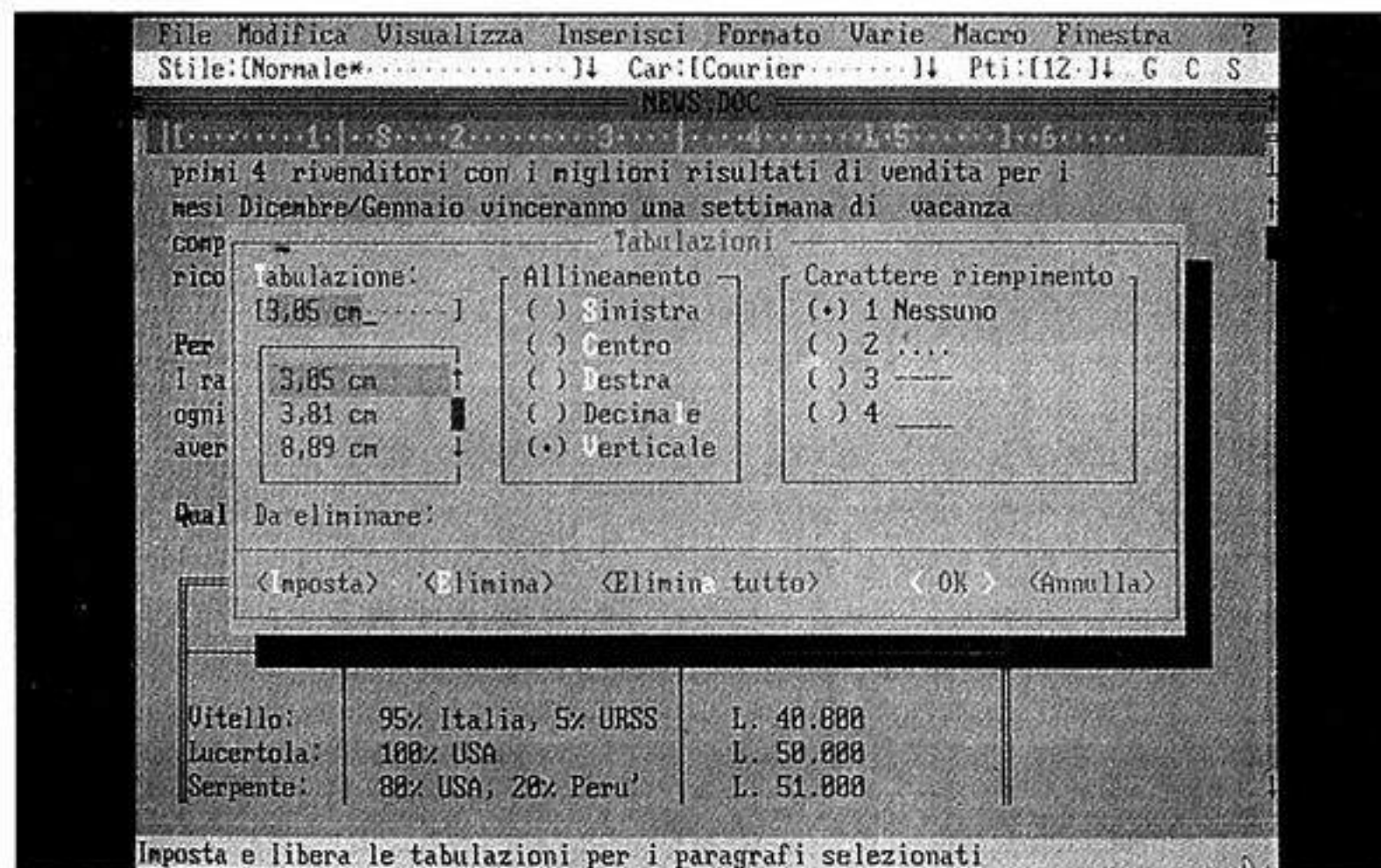
Oltre il w/p

Le notevoli potenzialità offerte dai moderni computer (e, parallelamente, dalle stampanti) consentono la manipolazione di un testo che, in pratica, può essere redatto dall'inizio (bozza ...correzioni ...stesura definitiva) alla fine (incolonnamento ...inserimento disegni e grafici ...didascalie ...riquadri) da **una sola persona** e da **un solo sistema** computerizzato.

Naturalmente **se** la persona è in grado di farlo, cosa che non è facile così come sembra.

Chi scrive queste note, ad esempio, ha iniziato a scrivere i suoi primi articoli servendosi di un text editor realizzato da sè (in Basic!) che in seguito, apportando alcune modifiche, fu trasformato in un word processor. A quei tempi, infatti, non c'era software specifico per il Commodore Pet (dotato, per giunta, di solo registratore) e bisognava arrangiarsi.

"Passato" al Vic 20 con drive 1540 acquistai la cartuccia Wordcraft (prezzo: circa 600 mila lire di allora...) con la quale il lavoro avanzò più speditamente. Con il C/64 (ed il w/p Easy Script) l'efficienza del lavoro raggiunse limiti incredibili.

Ora, per la stesura degli articoli, uso un Ms - Dos 80386, un programma di videoscrittura ed uno di impaginazione (con stampante laser...).

Non sono molti, però, a poter vantare una lunga esperienza nel campo della videoscrittura (raggiunta, per di più, lavorando nel campo specifico) nè si può pretendere, da un comune impiegato (cui devono esser destinati i programmi per "uso ufficio") il raggiungimento di un'esperienza analoga; se non altro perché la persona stessa dovrebbe esser distolta (allo scopo di conseguire tale esperienza) dal normale lavoro cui è destinato.

Dunque, da una parte c'è il desiderio di sfruttare al massimo le potenzialità offerte dai moderni sistemi informatici; dall'altra l'irrinunciabile esigenza di una rapida acquisizione dell'esperienza necessaria per sfruttare tali potenzialità (il tempo è denaro).

Il problema, almeno finora, si è risolto proponendo, nel campo della videoscrittura, **due** tipi fondamentali di pacchetti. Al primo appartengono i **word processor**; ai secondi i pacchetti di impaginazione (**DTP**).

Tutti noi abbiamo visto, almeno una volta, la corposità dei manuali di istruzione di un "semplice" word processor; e tutti noi, se siamo onesti ed equilibrati nei giudizi, sappiamo che per sfruttare adeguatamente le risorse offerte occorrono alcune dozzine di (intense) ore di esercitazione. Figuratevi un semplice impiegato, che dovrebbe (giustamente) servirsi del "sistema" hardware / software solo per lavorare (e che, altrettanto giustamente, non deve sentirsi in dovere di amare i calcolatori fino a smanettare perdutamente con essi). Se a questo impiegato, poi, chiediamo anche di imparare ad usare un DTP rischiamo le sue dimissioni per incompatibilità.

Chi non ha mai lavorato con **Ventura**, ad esempio, non può immaginare il tempo che ci vuole per impratichirsi dei comandi offerti dal potente DTP.

La strada normalmente seguita, quindi, non poteva essere che una sola: usare un w/p per il lavoro "normale" d'ufficio (servendosi, se del caso, dei vari stili, formattazioni e così via) allo scopo di produrre documenti dignitosi. Servirsi, invece, di servizi "esterni" all'azienda nel caso di realizzazione di depliant, brochure, manualistica varia o di altri documenti particolarmente impegnativi dal punto di vista dell'impaginazione.

L'anello di congiunzione

La **Microsoft**, con **Word 5.5** (disponibile anche in italiano) ha pensato bene di colmare il divario esistente tra i due tipi di pacchetti (w/p e DTP).

Word 5.5, infatti, è facile da usare come un w/p, ma consente impaginazioni degne di un vero DTP.

La filosofia del prodotto parte da un presupposto fondamentale: se un utente è abituato ad usare l'ambiente **Windows** (per il quale la Microsoft è giustamente famosa), un qualsiasi pacchetto che "assomiglia" al modo di operare di Windows è, di per sè, più semplice da usare rispetto a prodotti appartenenti alla stessa categoria, ma di provenienza diversa.

Spieghiamoci meglio: se un impiegato è stato abituato ad usare **Write** (il "mini" w/p disponibile "aprendo" Windows) per scrivere documenti anche complessi (Write è comunque un w/p di tutto rispetto); oppure a disegnare alcuni grafici servendosi di **Paintbrush** (il potente programma per disegnare, incorporato anch'esso in Windows); oppure, ancora, ad usare il data base (dal nome "**Schedario**") che gira "sotto" Windows; ebbene, questo stesso impiegato non troverà nessuna difficoltà ad usare il sistema di apertura e chiusura delle finestre (identico a qualsiasi programma Windows); oppure la funzione di taglia-e- incolla; oppure di registrazione e caricamento; e, non ultimo, il sistema di gestione di tutte le potenzialità via mouse, utilizzando i menu a tendina e le icone, magari personalizzate.

Chi, insomma, è abituato ad usare Windows e le sue potenzialità, può iniziare ad operare con Word 5.5 senza la minima difficoltà dal momento che si ritrova, quasi senza accorgersene, nello stesso ambiente in cui è abituato a lavorare.

In seguito, se vuole, può imparare a sfruttare tutte le potenzialità offerte da Word 5.5 rispetto (ad esempio) a Write, che sono, ovviamente, tantissime.

Operando con Word 5.5, quindi, si può passare gradualmente dall'uso di un semplice word processor allo sfruttamento totale di tutte le potenzialità offerte da un qualsiasi programma di impaginazione, compresa l'impaginazione su più colonne, l'uscita in PostScript o l'inserimento di immagini e grafici provenienti da altri "ambienti" in grado di produrli.

In un riquadro a parte, per chi se ne intende, riportiamo alcune delle potenzialità offerte dal potente pacchetto applicativo Microsoft.

Word 5.5, e qui chiudiamo, può rappresentare una validissima opportunità per quegli ambienti professionali in cui si comincia a sentire l'esigenza di un'impaginazione migliore rispetto a quella offerta da un comune w/p, anche se sofisticato; ma, nello stesso tempo, si manifesta una certa perplessità di fronte al notevole impegno richiesto per imparare ad usare uno dei DTP attualmente disponibili.

Un anello di congiunzione estremamente interessante, quindi, deve esser

Microsoft e dintorni

Pochi sanno che la Microsoft non si limita ad offrire un semplice servizio di vendita; chi, infatti, acquista un prodotto originale Microsoft ha diritto ad una serie di servizi, tutti molto interessanti per l'utente.

Il centralino della Microsoft (Tel. 02/26.91.21) è dotato di ben 24 linee, mentre 20 linee supplementari sono a disposizione dei servizi diretti, cioè informazioni commerciali pre-vendita (02/26.90.1.359) e gestione ordini post-vendita (02/26.90.1.333).

La "linea" che più interesserà i nostri lettori è forse la **Hot Line Tecnica** che si divide tra applicativi e linguaggi, tra cui Excel, Word, Works, Windows, Cobol, Basic e QuickBasic ecc. (02/26.90.1.351); linguaggi, prodotti network e toolkit di sviluppo, tra cui gli altri linguaggi (C, Pascal, Masm, Fortran, Lan Manager, Sql Server, Os/2) ed i prodotti per le reti locali (02/26.90.1.354).

Per maggiori informazioni si consiglia di contattare direttamente:
Microsoft S.p.a.
Milano Oltre
Palazzo Tiepolo
Via Cassanese, 224
20090 Segrate (Mi)
Tel. 02 / 26.91.21
Fax 02 / 21.07.20.20

considerato Word 5.5 che, al vantaggio di una manualistica in italiano, offre la tipica semplicità dell'ambiente Windows unita alle potenzialità riscontrabili in un moderno DTP.

Se, infine, si considera che il prezzo di vendita al pubblico (**L. 950 mila**) è più vicino a quelli, tipici, della fascia dei word processor, più che a quelli di un Desk Top Publishing, la faccenda diventa molto, molto interessante...

Il mondo esterno

Una delle caratteristiche che rende Word 5.5 un prodotto realmente professionale è la disponibilità del pacchetto in tre diversi ambienti, detti in gergo "**piattaforme**". Ci riferiamo all'**Ms - Dos** (o meglio, Windows), all'**OS/2** ed al mondo **Macintosh**.

Le tre versioni sono praticamente identiche per l'utente finale, il quale, se si limita ad esaminare le schermate ed il modo di lavorare di Word 5.5, non è quasi in grado di individuare differenze operative, se non... guardando il computer che sta usando in quel momento.

Questo particolare consente di usare Word 5.5 in quegli ambienti professionali (pensate ai grossi uffici con una ventina di dipendenti) in cui, per una serie di scelte, operano computer fondamentalmente diversi come hardware. I file prodotti, ad esempio, da un computer Ms - Dos, possono essere "ripresi" ed elaborati da un Macintosh, grazie al formato "unico" del prodotto Microsoft.

Inutile soffermarsi sulle potenzialità di un simile prodotto operando in rete, soprattutto se si considera che Word 5.5 supporta il formato **RTF** (**R**ich **T**ext **F**ormat) che consente la "condivisione" di files con piattaforme e sistemi operativi diversi.

Per ciò che riguarda gli help utili per l'utente finale, ricordiamo che Word 5.5 supporta **Thesaurus** in italiano (funzione che consente la ricerca dei sinonimi), in grado di spaziare tra **16 mila** parole chiave ed oltre **150 mila** vocaboli.

Più di un Word processor

I word processor della vecchia generazione consentono limitate *pre-view* dell'aspetto finale del documento.

Ciò significa che, prima di mandare in stampa il documento digitato, è possibile richiamare un'opzione che visualizza, su schermo, il foglio che verrà riprodotto su carta.

Alcuni moderni w/p, in verità, consentono tale opportunità, ma spesso non è consentito ottenere ingrandimenti di porzioni del foglio, con la conseguenza di non potere avere una chiara idea dell'aspetto finale.

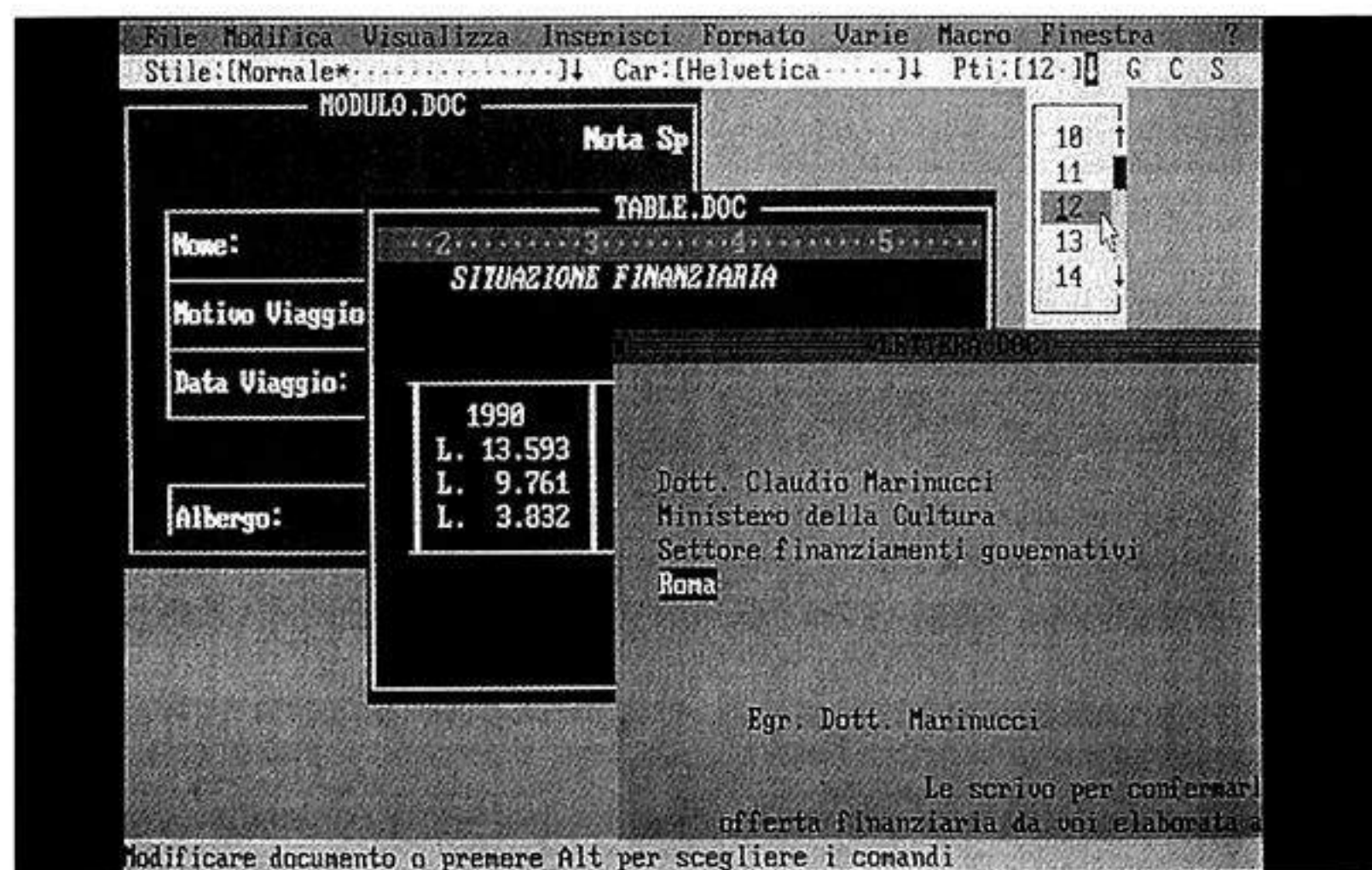
Con Word 5.5, invece, non solo è possibile esaminare un documento nei minimi dettagli, ma addirittura questo viene riprodotto a colori e corredato di tutti i settaggi imposti durante la digitazione (formattazione, stili e font particolari, eccetera) in accordo con i più rigorosi dettami di un sistema WYSIWYG che si rispetti (**W**hat **Y**ou **S**ee **I**s **W**hat **Y**ou **G**et: ciò che vedi è ciò che ottieni).

Le principali difficoltà della stesura di un testo qualsiasi vengono risolte in modo semplice ed efficace: la gestione semplificata delle **tabelle**, l'imposizione di **stili** diversi in altrettante porzioni del documento, il supporto di quasi tutte le stampanti **laser**, l'indicazione automatica delle **correzioni**, i collegamenti dinamici con i **fogli elettronici**, la possibilità di effettuare operazioni di taglia e incolla di testi e **grafici** provenienti da altri programmi che girano sotto Windows, il completo supporto del **mouse**, la personalizzazione di **macro** istruzioni, la gestione contemporanea di più documenti, la possibilità di quattro metodi di visualizzazione **pre-view**, il completo supporto di **rete** e di posta elettronica, la possibilità di realizzare **menu** personalizzati: sono soltanto alcune delle caratteristiche sulle quali non è possibile intrattenersi più di tanto per ovvie questioni di spazio.

Il principale vantaggio offerto da Word 5.5, tuttavia, non è tanto costituito dalle straordinarie **potenzialità** offerte dal prodotto; quanto, a nostro parere, dalla velocità di **apprendimento** necessaria per raggiungere determinati risultati.

Spieghiamoci meglio, considerando un utente "medio" (già esperto di un qualsiasi word/processor, di marca non Microsoft) che venga invitato ad utilizzare un programma di impaginazione (DTP) servendosi dei soli manuali di istruzione allegati alla confezione.

L'obiettivo da raggiungere è quello di imparare ad impaginare un testo ipotizzando una situazione reale abbastanza frequente: due colonne di testo impaginato in modo da comprendere qualche frame (occupato da immagini provenienti da ambienti disparati), varie tabelle (formate da almeno una dozzina di righe e cinque colonne), diversi stili e font e così via.

Il tempo necessario per l'autoistruzione può risultare minore se, invece di servirsi di un DTP, l'utente decida di imparare ex-novo Word 5.5 per raggiungere gli stessi risultati.

di Alessandro de Simone

PostScript, alla ricerca del linguaggio perduto

La mancata lungimiranza dei progettisti di stampanti ha costretto il ricorso a tecnologie di stampa "alternative"

Il classico senno di poi consente, per nostra fortuna(!), di parlar male delle persone che, nel passato, si sono comportate esattamente come ci saremmo comportati noi, al loro posto.

Quando si compivano i primi passi della scrittura meccanizzata, infatti, le **telescriventi** (termine ora arcaico ma che, per quei tempi, aveva un sottile sapore fantascientifico) non erano altro che macchine da scrivere in cui, al posto dei tasti, erano sistemate elettrocalamite in grado di imprimere la spinta necessaria al martelletto.

Su ognuno di questi, come è noto, era presente il carattere da stampare che lasciava la propria traccia sul foglio di carta non appena l'elettrocalamita riceveva l'impulso che le competeva.

Dalla macchina da scrivere elettrica (gestita dai fantozziani *diti* umani) alla macchina da scrivere governata dal computer (a quei tempi a valvola, poi a transistor ed in seguito a circuiti integrati) il passo fu breve: fu sufficiente fabbricare un'interfaccia in grado di inviare alla telescrivente gli impulsi che spingevano i singoli martelletti.

Dal momento che, anche a quei tempi, il computer era enormemente più veloce della macchina da scrivere, nessuno si preoccupò più di tanto di ottimizzare il trasferimento dei dati; questi, in ogni caso, erano stampati ad una velocità incomparabilmente superiore a quella che si poteva ottenere da comuni impiegati umani.

Se un foglio dattiloscritto, insomma, veniva fuori dopo un minuto (velocità pari a una riga al secondo) nessuno protestava, anzi.

Particolare di una stampante laser (Epson EPL - 7100)
Una stampante laser, predisposta per il sistema PostScript, consente la modifica della periferica mediante la semplice aggiunta della scheda contenetè la circuiteria consentita dalla Adobe.

In seguito, con l'invenzione delle stampanti ad aghi, la velocità aumentò ancora, grazie alla riduzione dell'inerzia meccanica dovuta alla minore massa rappresentata dai singoli dot (aghi della stampante) rispetto ai più corposi martelletti (stampanti a **margherita**) o a **sfera** (brevetto IBM).

In ogni caso, l'enorme velocità di stampa, se confrontata sempre con l'operatore umano, aggiunta alla possibilità di ridefinire i caratteri (cosa impossibile con le tecniche precedenti) fece passare in secondo piano una pecca che, per quei tempi, incominciava a manifestarsi in tutta la sua potenzialità negativa.

La grafica

Le stampanti ad aghi, grazie alla loro stessa costituzione fisica, sono in grado di stampare un singolo punto in una posizione qualsiasi del foglio di carta.

Anche con gli schermi grafici è possibile far apparire acceso (oppure spento) un singolo **pixel** del video. Forte di questa banale analogia, a qualcuno venne in mente di trasferire, pari pari, i pixel presenti sullo schermo, direttamente sul foglio di carta di una stampante ad aghi.

Tutti noi conosciamo la possibilità di ottenere **hard copy** con qualsiasi computer (ricordate il C/64?), utilizzando un'infinità di routine, scritte prevalentemente in linguaggio macchina.

Nonostante la notevole efficienza delle routine e l'aumento della velocità dei computer, una schermata in alta risoluzione (e, badate bene, solo in bianco e nero), richiedeva (e richiede) una dozzina di minuti per la sua stampa completa. Notate che stiamo parlando di schermate di 640 x 200 pixel, vale a dire di "appena" **128 mila** pixel (80 colonne x 25 righe). Se, poi, vogliamo anche il colore, le attuali tecnologie di stampa (ci riferiamo ai nastri a colori) richiedono tempi almeno tripli, dal momento che ogni pixel deve essere "processato" tre volte prima di assumere il colore voluto su carta.

Quale è il motivo di una simile lentezza operativa, ormai inaccettabile anche lavorando con stampanti ad aghi veloci ed accontentandosi di schermate in bianco e nero?

Tutto deriva dal fatto che l'interfaccia di solito utilizzata dalle stampanti (centronics, parallela; oppure, peggio, seriale), consente l'invio di pochi bit per volta; questi, giunti nella stampante, vengono elaborati in modo da governare opportunamente sia l'avanzamento della testina, sia l'impatto di uno, o più, aghi.

Il collo di bottiglia, dunque, è rappresentato anche dall'interfaccia, o meglio dalle caratteristiche del trasferimento dei files.

Spieghiamoci meglio: quando si realizzarono le prime interfacce, le esigenze di allora si limitavano alla gestione di pochi bytes per pagina.

Dal momento che il mercato richiedeva una standardizzazione del formato, quello **Centronics** risultò il più affidabile ed economico.

La conseguenza fu che tutti i produttori, di hardware e di software, si buttarono a capofitto sullo sfruttamento totale di tale standard, che divenne, di fatto, quello più diffuso nel campo dell'informatica.

La stampante laser Epson EPL - 7500

L'aspetto esterno di una stampante PostScript non differisce da modelli analoghi che non prevedono il linguaggio della Adobe. La meccanica, di solito, è identica nei vari modelli e la risoluzione raggiungibile, di conseguenza, rigorosamente eguale. Ciò che cambia, ovviamente, è la velocità di riproduzione raggiungibile nei due casi.

Oggi, però, le esigenze sono cambiate: e non ci riferiamo soltanto alla necessità di stampare schermate in alta risoluzione (che, come abbiamo visto, richiedono, nei casi più semplici, l'invio di 128 mila bit, oltre ai byte di controllo) ma anche alle nuove potenzialità offerte dalle nuove stampanti a descrizione di pagina.

Le stampanti laser

Il limite del singolo **dot** (sempre più piccolo, sempre più "definito" nelle moderne stampanti ad aghi) non ha nulla a che fare, tuttavia, con le tecnologie di stampa alternative, tra cui spiccano quelle fotografiche, per ora irraggiungibili da tecnologie non esplicitamente fotografiche.

Le stampanti laser, come anche alcune stampanti a **getto di inchiostro**, sono in grado di gestire **300 x 300** puntini elementari per pollice quadrato. Ciò significa che, in una superficie di nemmeno 6.5 centimetri quadrati, è possibile stipare **90 mila** puntini (cioè circa 14 mila punti per centimetro quadro). Un normale foglio di carta di formato A4 (cm. 21 x 29.7), dunque, può "ospitare" circa **9 milioni** di dot, se "trattato" da una stampante laser.

Riuscite ad immaginare che significa trasferire, da computer a stampante, una simile massa di dati attraverso il collo di bottiglia di un'interfaccia Centronics in grado, a malapena, di stampare una schermata grafica di 128 k in una dozzina di minuti?

Il problema dei fabbricanti di stampanti (e di software), quindi, è duplice: da un lato è indispensabile salvaguardare gli investimenti effettuati dai fabbricanti sugli standard più diffusi al mondo; dall'altro, invece, pur volendo restare nell'ambito dei trasferimenti bit per bit (decisamente inefficiente, soprattutto pensando alle possibilità grafiche dei computer futuri), la massa di dati in gioco è talmente elevata che l'eventuale aumento della velocità di trasferimento potrebbe dimostrarsi inefficace dopo qualche anno di dignitosa vitalità.

Mondo grafico

Il problema, comunque, rimane legato a diversi fattori concomitanti: da un lato c'è la necessità di adeguare il parco stampanti degli utilizzatori; tale esigenza

è sentita sia da questi ultimi che (soprattutto!) dai fabbricanti di stampanti.

I primi, infatti, non sono soddisfatti della qualità grafica offerta dalle stampanti ad aghi (piuttosto bruttina, diciamo la verità); i secondi, inoltre, si sono accorti che gli utilizzatori sono restii a sostituire la propria stampante perché (diciamo ancòra la verità) la differenza di qualità, passando da una 9 aghi ad una 24 aghi, non è così rilevante da consigliarne la sostituzione.

In media, infatti, un utilizzatore finale, che intenda aggiornare il proprio sistema informatico, è propenso a sostituire il computer con uno più veloce, al quale continua a collegare la vecchia stampante ad aghi.

La molla che può smuovere il mercato delle printer può quindi essere costituito da una qualità **nettamente** superiore, ottenibile, oggi come oggi, solo da stampanti laser o a getto di inchiostro.

Non si può, tuttavia, convincere l'utente all'acquisto di una stampante laser se questa, pur fornendo una qualità eccellente, richiede almeno mezz'ora per la stampa di una schermata grafica.

Occorre uno, o più, *"plus"*, vale a dire una pluralità di migliorìe che rendano definitivamente obsoleta ed inefficiente una qualsiasi stampante ad aghi.

Limiti video

Si potrebbe obiettare che la tecnologia laser (capace di 300 x 300 dot, cioè di circa 9 milioni di dot su un foglio "normale") è sprecata per riprodurre schermate, anche se queste sono realizzate con le più sofisticate schede grafiche attualmente disponibili.

Oggi, infatti, una scheda grafica da 1024 x 768 pixel, sufficientemente venduta in massa nel mondo Ms - Dos, richiede la gestione di poco più di **786 mila** pixel.

Tale considerazione, però, cade miseramente non appena si pensa che ciò che appare su video potrebbe rappresentare (se il software lo consente) solo una **porzione** del foglio di carta da riprodurre. Ad esempio possiamo immaginare che, sul foglio di carta, potrebbero esser trasferite quattro (sei, otto...) schermate grafiche, **ognuna** con risoluzione di 1024 x 768 puntini elementari.

Inoltre dobbiamo ricordare che alcune immagini, come le fotografie provenienti da uno scanner, sono visualizzate su schermo con una risoluzione approssimativa (dovuta, appunto, alla limitata risoluzione del video) ma, in realtà, sono memorizzate in un formato, molto più preciso, che viene evidenziato all'atto del trasferimento su una periferica in grado di sfruttarne la risoluzione. Non è un mistero per nessuno, insomma, che un'immagine scannerizzata può apparire più nitida su carta (laser) che su video.

I moderni software di impaginazione (**DTP**, Desk Top Publishing), poi, impegnano in modo massiccio i sistemi di riproduzione su carta; basta pensare al titolo dell'articolo che state leggendo: se fosse possibile determinare facilmente l'area "annerita" rappresentata delle semplici parole del titolo *(PostScript, alla ricerca del linguaggio perduto)*, vi accorgereste che i centimetri quadri (ed i corrispondenti K dot) si sprecano.

L'uovo di Colombo

Ma perché mai, qualcuno si chiese, bisogna trasferire una schermata grafica bit per bit? Se, ad esempio, vogliamo tracciare un quadrato nero di 3 pollici per lato (totale: 9 pollici quadri, 810 mila dot alla risoluzione laser di 300 x 300) è inutile inviare, appunto, 810 mila "segnali".

Dovrebbe esser sufficiente inviare alla stampante solo alcune informazioni: posizionamento dello spigolo superiore sinistro (rispetto al foglio di carta), lunghezza del lato verticale e del lato orizzontale, indicazione di "riempimento" della figura.

Le poche decine di dati possono esser trasferire in un battibaleno, anche per mezzo dell'interfaccia Centronics, all'interno della stampante laser; qui, ad attenderli, sarà presente una vera e propria **unità di elaborazione** (un vero computer, insomma) che non solo è in grado di elaborare per proprio conto le informazioni ricevute (gestendo la testina laser in modo adeguato) ma consentendo al computer che ha inviato i dati di essere subito disponibile per altre elaborazioni.

In pratica, è come se, lavorando in **Gw - Basic**, decidessimo di disegnare un quadrato con l'istruzione **Line** (corredata di pochissimi parametri) invece di tracciare, pixel per pixel, l'intera superficie del quadrato voluto.

Riepilogando: la tecnologia laser, grazie alla notevole risoluzione che offre, impone la gestione di milioni di puntini elementari, sia per stampare testi (anche un "piccolo" carattere alfanumerico può esser formato da centinaia di dot) sia grafici. Il collo di bottiglia è rappresentato sia dalle interfacce attualmente disponibili (incapaci di smistare milioni di byte in modo efficiente) sia dalla eccessiva lunghezza dei files che, anche se si risolvesse il problema della velocità di trasferimento, occuperebbero addirittura intere porzioni di dischi rigidi.

Il problema, quindi, si risolve ricorrendo ad un linguaggio interprete che sia in grado di elaborare, con poche istruzioni, tanto i caratteri alfanumerici, quanto le immagini grafiche. La necessità di usare un tale sistema si manifesta nelle stampanti ad alta risoluzione. Il sistema di stampa viene definito, per tale motivo, *"a descrizione di pagina"* dal momento che una pagina viene elaborata (*descritta*, appunto) mediante le istruzioni di un software specifico.

PostScript

La **Adobe Inc.** (che si legge *Adòbi*, e non A'dob) sviluppò un linguaggio (giunto oggi alla versione 52.3) in grado di soddisfare le esigenze dei fabbricanti di stampanti a descrizione di pagina.

Si tratta di software sofisticato, difficilmente duplicabile dal momento che, per funzionare, è indispensabile che sia presente su Rom e che sia supportato da un'unità di elaborazione presente all'interno della stessa stampante.

In pratica, anche se riuscissimo a copiare le Rom presenti in una stampante PostScript, non potremmo inserirle nella nostra stampantina ad aghi (dove, del resto?).

Il motivo della impossibilità (legale, più che tecnica) di copiare il software, ha consentito, alla Adobe, di sfruttare adeguatamente il brevetto e di farsi pagare i diritti dai fabbricanti di printer che intendano proporre lo standard PostScript ai propri utenti.

Si pensi che una cartuccia originale PostScript, in grado di trasformare una "semplice" stampante laser (ovviamente predisposta per l'operazione) in una printer PostScript compatibile, può raggiungere la cifra dei tre milioni di lire!

Qualcuno (sempre quello di prima) potrebbe obiettare che linguaggi dotati di istruzioni grafiche (Basic, Pascal, C) era-

Prova di stampa; questo è un carattere ***Helvetica Narrow corpo 16***

Prova di stampa; questo è un carattere Times

Prova di stampa con stampante dotata di sistema PostScript ("vero" o simulato). Questo è un carattere **Helvetiva "corpo" 9.0**

Prova di stampa; questo è un carattere ***Helvetica Narrow corpo 16***

Prova di stampa; questo è un carattere Times

Prova di stampa con stampante dotata di sistema PostScript ("vero" o simulato). Questo è un carattere **Helvetiva "corpo" 9.0**

Prova di stampa; questo è un carattere ***Helvetica Narrow corpo 16***

Prova di stampa; questo è un carattere Times

Prova di stampa con stampante dotata di sistema PostScript ("vero" o simulato). Questo è un carattere **Helvetiva "corpo" 9.0**

Prova di stampa; questo è un carattere ***Helvetica Narrow corpo 16***

Prova di stampa; questo è un carattere Times

Prove di stampa

I quattro "rettangoli" contengono altrettante immagini rigorosamente identiche (in origine). Sono state realizzate usando esclusivamente il programma di impaginazione **DTP Ventura** e riprodotte su quattro periferiche diverse: una stampante ad aghi ed una laser **non** PostScript (gestite entrambe dal pacchetto **Freedom of Press**); una stampante laser PostScript **originale Adobe;** un'unità di fotocomposizione capace di 1200 x 1200 dot. La differenza è evidente tra le diverse tecnologie di stampa. La differenza tra le due stampanti laser è riscontrabile solo in termini di tempo: l'unità originale PostScript è una scheggia se paragonata a quella offerta in emulazione.

no già disponibili quando si manifestò l'esigenza di "descrivere" una pagina; perché mai inventarsi un nuovo (per di più costosissimo) linguaggio?

I motivi sono numerosi; anzitutto PostScript non è solo un linguaggio, ma un vero e proprio "sistema", notevolmente versatile. PostScript lavora sia su stampanti sia su schermi sia su apparecchi basati su tecnologie fotografiche; un'unità di fotocomposizione raggiunge una definizione di **1200 x 1200** punti per pollice quadro e rappresenta quanto di meglio oggi si può avere in termini di riproduzione su carta (una definizione superiore sarebbe inutile: l'occhio umano non è in grado di accorgersi degli eventuali miglioramenti di definizione). PostScript, inoltre, gestisce il colore ed è predisposto, per quanto possibile, con le innovazioni future nel campo delle riproduzioni di immagini e testi. Una parte considerevole dei ricavi delle licenze d'uso, infatti, viene destinato alla ricerca ed alle migliorie da apportare, sia nel campo del software che in quello dell'hardware.

Quasi tutte le software house professionali, inoltre, offrono programmi che consentono la gestione di stampanti PostScript, soprattutto per pacchetti applicativi grafici (si pensi all'eccellente **AutoCad**) e di DTP (basterà citare **Ventura** e **PageMaker**).

Un "semplice" Pascal, C, o Basic, quindi, era inadeguato allo scopo; oppure avrebbe costretto a continue migliorie ed adattamenti, pregiudicando la diffusione di uno standard affidabile ed universale.

Ancòra grafica

Un file PostScript, dunque, non è altro che un **programma** che, opportunamente interpretato (dopo essere stato trasferito nella stampante predisposta allo scopo) riprodurrà sul foglio di carta (o schermo, se si tratta di un file destinato ad "uscire" sul video di un terminale PostScript) l'immagine desiderata.

La velocità con cui quest'ultima viene generata non è dovuta solo al ridotto numero dei bytes inviati alla periferica (rispetto ad un'immagine bit-mapped), ma anche, o soprattutto, alla gestione hardware della periferica stessa.

In altre parole, l'algoritmo che viene attivato, ad esempio, per riprodurre una vocale "**A**" maiuscola che misura due centimetri di base per quattro di altezza, è il vero responsabile dell'elevata velocità di gestione del "pennino" (raggio laser, nel caso specifico di tali stampanti) di cui è dotata la periferica. Allo stesso modo, un'elevata velocità di gestione è riscontrabile nel tracciare segmenti, archi di cerchio, pattern di riempimento ed altri "oggetti" grafici (o riconducibili a forme grafiche, come i caratteri alfanumerici) che risultino facilmente, si fa per dire, parametrizzabili.

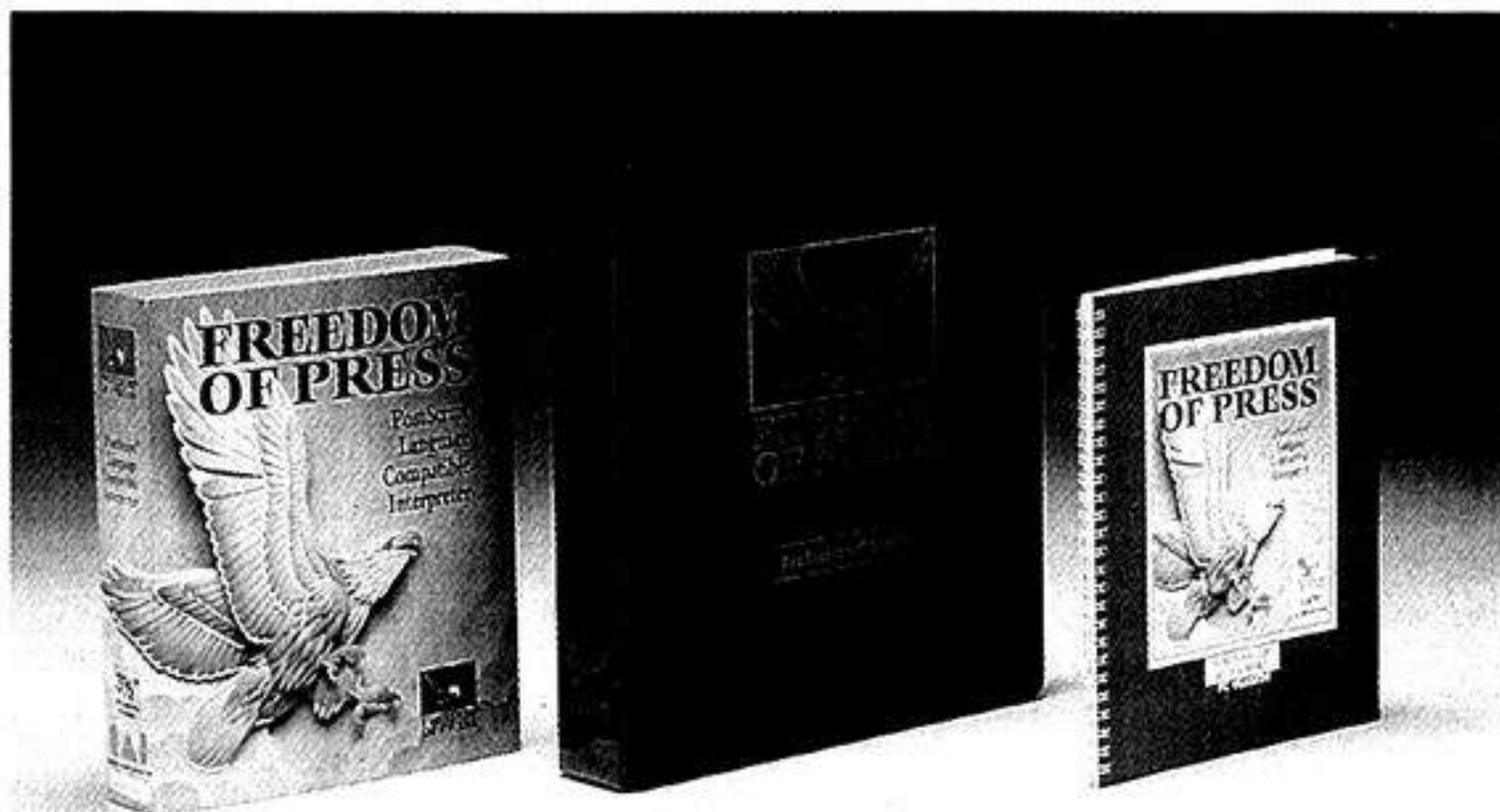
Una fotografia o un disegno, scannerizzato o realizzato a mano libera con uno dei tanti tool grafici, purtroppo, non può essere parametrizzato; non è possibile, ad esempio, scrivere un algoritmo che, in un ritratto fotografico, sia in grado di individuare il contorno di un occhio (e "definirlo" cerchio) o i segmenti curvi che costituiscono i peli delle sopracciglia (e stabilirne i parametri assimilando, queste ultime, ad archi di ellisse).

Freedom of Press

Freedom of Press (FOP) è un pacchetto (distribuito da **Channel Srl, Il Girasole, Pal. 3/05 A, c.a.p. 20084 Lacchiarella, Milano; tel. 02 / 90.09.17.73**) che gira su computer Ms - Dos dotati di almeno 2 mega di Ram. In pratica FOP è in grado di "leggere" un qualsiasi file PostScript utilizzando le potenzialità elaborative del computer (anziché della stampante, come in un "vero" sistema PostScript) per la riproduzione del foglio di carta.

I vantaggi sono evidenti: anzitutto è possibile usare anche stampanti ad aghi, tradizionalmente prive del famoso sistema a descrizione di pagina; il secondo vantaggio (potevate dubitarlo?) è economico: il prezzo del pacchetto è infatti **inferiore** alle 800 mila lire. Lo svantaggio più evidente è rappresentato dalla minore velocità operativa: circa **cinque minuti** per una pagina "media" della rivista che hai in mano (con stampante ad aghi), che scendono a meno della metà se si utilizza una laser (ovviamente non PostScript); un altro svantaggio, pur se di minore entità, è rappresentato dal fatto che, durante l'elaborazione, il computer è ovviamente impegnato ad... elaborare la pagina.

Si tenga presente che, per usare FOP con altri pacchetti applicativi che generano files di tipo PostScript (come Ventura, ad esempio), è necessario "uscire" dall'applicativo, "entrare" in FOP e gestire il file precedentemente memorizzato.

A onor del vero è possibile, con alcuni pacchetti, evitare queste lungaggini. Rimane il fatto, però, che anche "restando" in Ventura, ad esempio, l'operatore è impossibilitato ad operare con il computer finché la stampante non termina di svuotare l'ultimo buffer.

In questi casi, ahinoi, l'unico modo di gestire il file è quello di trasformare la foto in un'immagine bit-mapped, con conseguente spreco di bit, di occupazione di spazio su disco (pensate anche all'eventuale invio via modem...) e di trasferimento, e successiva elaborazione, su stampante. L'unico modo per risparmiar tempo, in casi come questo, consiste nel ricorso a tecniche di compressione dei files contenenti immagini di tal tipo (bit-mapped); in ogni caso, comunque, questo artifizio può consentire un certo risparmio in termini di tempo di trasferimento da computer a terminale: il tempo di elaborazione, inevitabilmente legato al numero di bit che costituiscono l'immagine, rimane elevato.

Le immagini bit mapped

Per bit - mapped si intende un'immagine considerata come un insieme rettangolare di puntini elementari, accesi o spenti a seconda se devono comparire sulla superficie oppure no.

Se l'immagine è a colori, oltre al bit che indica se un certo punto deve comparire, o meno, devono esser presenti altri bit in grado di stabilire il colore del punto stesso oppure, se questo dovesse risultare "spento", quello del "fondo" che compare in sua assenza. In pratica, anche un unico punto nero presente all'interno di un immenso rettangolo bianco richiede una gran quantità di bit per essere definito.

In quest'ultimo caso, però, assumono un certo interesse le tecniche di gestione delle immagini bit-mapped che fanno parte integrante del software di gestione o della periferica (o di entrambe).

Per non appesantire il concetto ricorreremo ad un banale esempio pratico.

Le prime stampanti costringevano la testina di scrittura ad andare dall'estrema sinistra all'estrema destra del rigo di carta anche nei casi in cui c'era la necessità di stampare un solo carattere posto a sinistra del foglio stesso.

In seguito, per risparmiare tempo, furono inventati due artifici, il primo s/w ed il secondo h/w. Grazie al primo, infatti, se la stampante si "accorgeva" di dover stampare alcuni righi "a vuoto" (senza stampar nulla, ad esempio, tra due capoversi), la testina non si spostava orizzontalmente, ma il rullo faceva avanzare la carta di tanti righi quanto era necessario; grazie al secondo accorgimento, invece, la testina di stampa, giunta all'estrema destra del rigo stampato, non tornava subito indietro (senza stampar nulla) ma, dopo aver fatto avanzare il rullo di un rigo, veniva utilizzata per stampare il rigo successivo (ovviamente al contrario, cioè dall'ultimo carattere al primo) partendo da destra. In questo modo, eliminando i tempi morti, la velocità globale della periferica aumentava in modo considerevole.

Gli sforzi dei fabbricanti di stampanti vergono tutti, oggi, nell'ottimizzare i percorsi dei "pennelli" sia a livello h/w che s/w al punto di suggerire, ai fabbricanti di microprocessori e circuiti integrati, le caratteristiche da inserire nei chip del futuro.

I files PostScript

Se avete occasione di esaminare un file PostScript, vi accorgerete che questo è un normalissimo **file ASCII**, nè più nè meno di un qualsiasi file sorgente in Turbo Pascal o in QuickBasic: potete esaminarlo con il comando Type del Dos ed inviarlo su stampante per studiarlo con più attenzione.

Il linguaggio PostScript, infatti, è un linguaggio ad alto livello che consente di inserire commenti, istruzioni di salti, utilizzo di procedure particolari, cicli iterativi e così via.

Se disporrete (o meglio, *quando* disporrete...) di una stampante PostScript, potrete scrivere da soli il vostro bravo programmino che stamperà rettangoli, cerchi, barre sfumate, caratteri di ogni grandezza, orientati in modo qualsiasi e tante altre "figure" difficilmente stampabili con altri sistemi.

I libri sul linguaggio PostScript non sono così semplici da trovare in libreria a causa del limitato interesse per il pubblico *di massa:* scrive in PostScript chi possiede, anzitutto, una stampante adeguata; poi chi sviluppa software di notevole livello.

Ai curiosi diremo solo che non è un linguaggio più complesso del Turbo Pascal, ma che richiede molto tempo e pazienza (oltre ad una considerevole mole di fogli di carta per le prove...) prima di impratichirsi adeguatamente.

Per avere un foglio di carta in formato PostScript, comunque, **non** è assolutamente necessario imparare a programmare, ci mancherebbe altro!

Tutti i software professionali (o di un certo livello, il che è lo stesso) offrono la possibilità di **generare automaticamente files in formato PostScript**. Dopo aver disegnato una figura geometrica con il famoso AutoCad, ad esempio, è possibile registrarla su disco; "usciti" da AutoCad, in seguito, potremo esaminare con calma ciò che il programma ha prodotto. Non illudetevi, comunque, di trovarvi di fronte ad un file molto breve: anche se avete disegnato un semplice quadratino, e nient'altro, il file PostScript conterrà una miriade di informazioni, apparentemente inutili, in grado di settare correttamente la stampante. La stessa cosa dicasi per i files PostScript generati da un banale Word Processor.

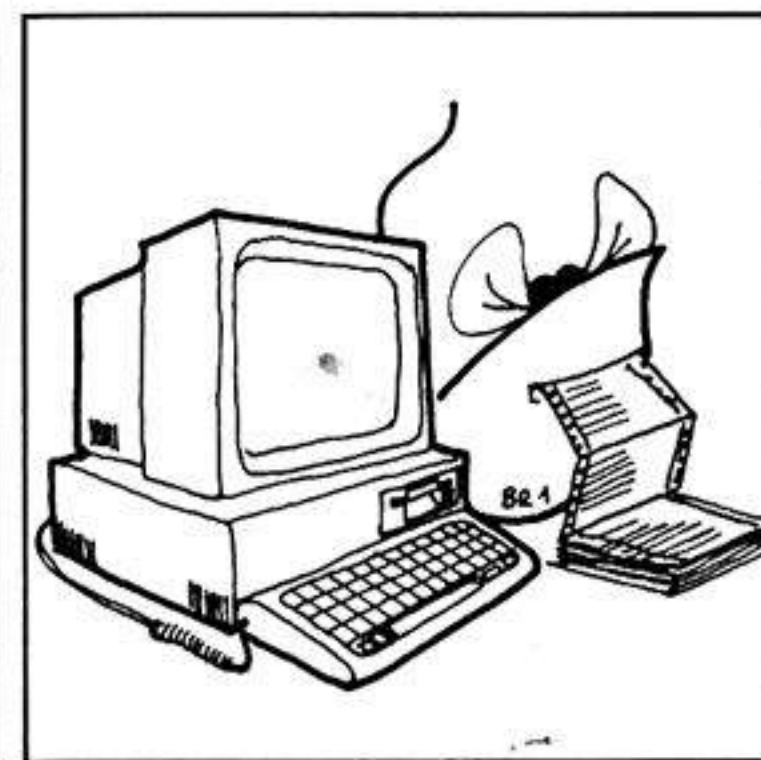
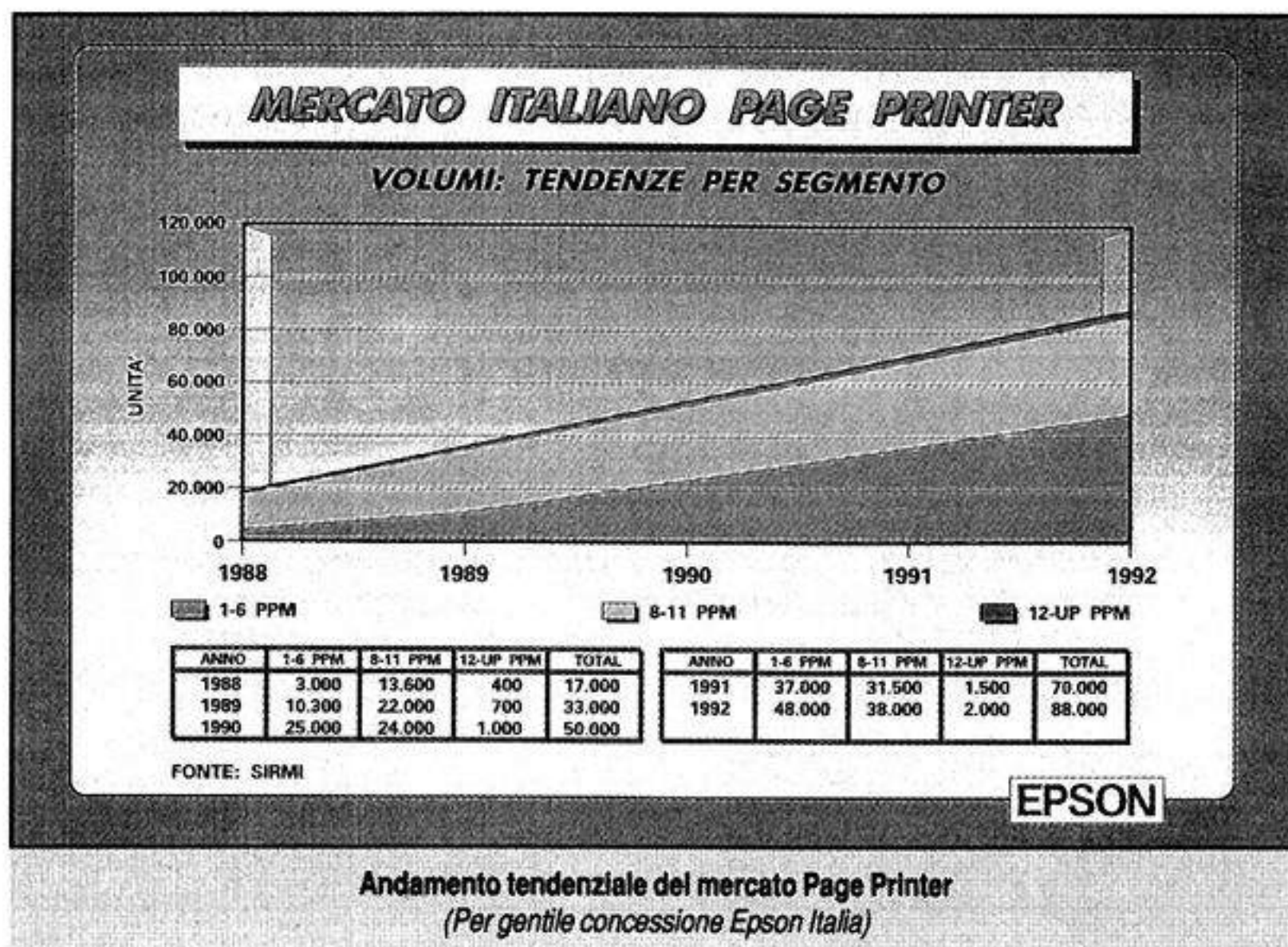
Emulazione PostScript

Alla fine della chiacchierata qualcuno (sempre quello di prima) si sarà accorto (speriamo...) che un linguaggio orientato alla descrizione della pagina non vive solo nelle stampanti laser o, comunque, nei mostri che consentono risoluzioni altissime rispetto ad una stampante ad aghi.

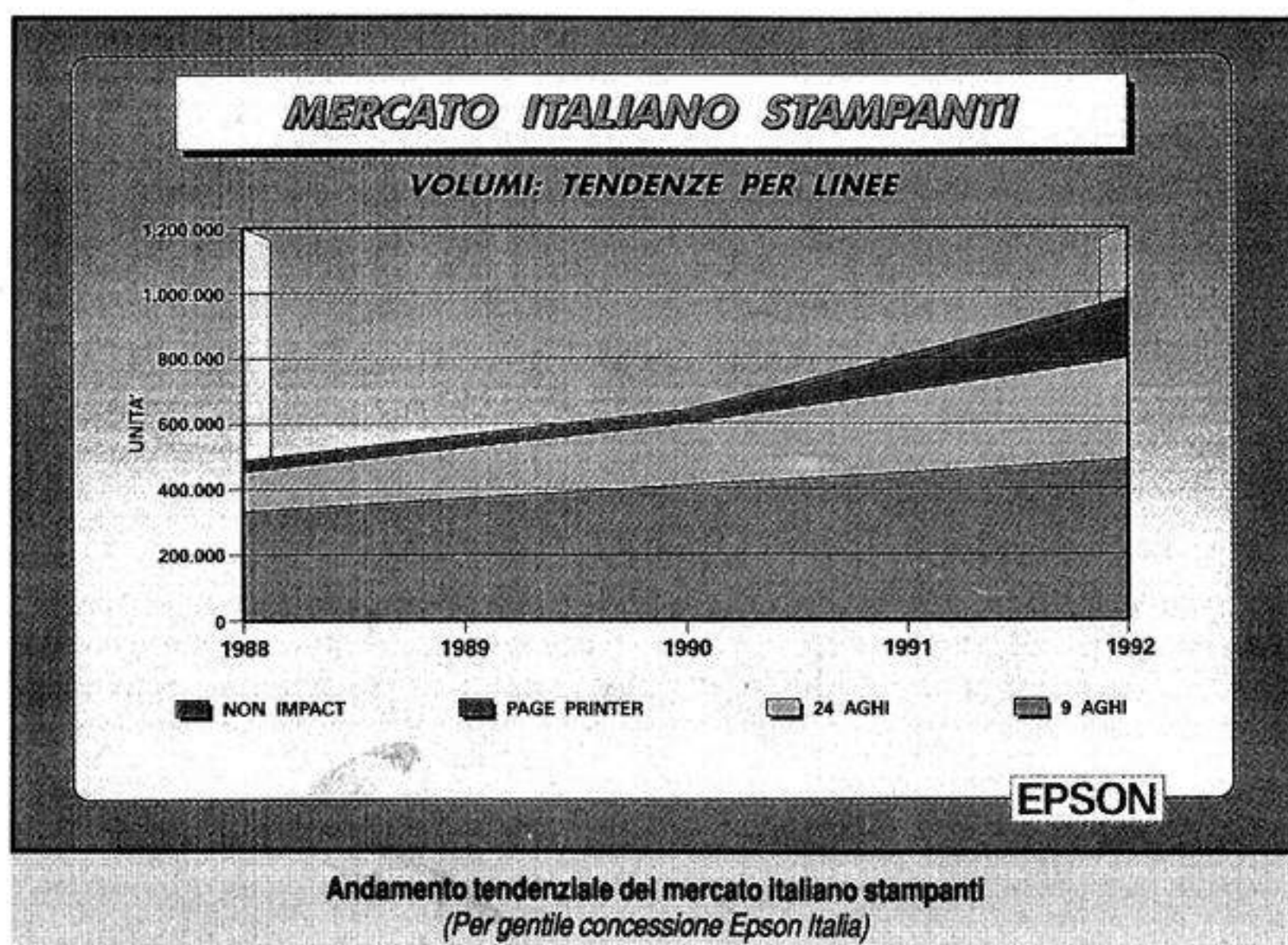
Il linguaggio PostScript, infatti, può in teoria (e, soprattutto, in pratica) funzionare egregiamente anche su **stampanti ad aghi**.

E' ovvio che, in quest'ultimo caso, l'aspetto finale del foglio di carta risentirà della modesta risoluzione offerta dalla printer ad aghi; se, però, sovrapponiamo due fogli identici, contenenti testo e grafica (generati con linguaggio PostScript) ma il primo realizzato con stampante laser ed il secondo con stampante ad aghi, noteremo che gli "ingombri", come i po-

ANNO	1-6 PPM	8-11 PPM	12-UP PPM	TOTAL
1988	3.000	13.600	400	17.000
1989	10.300	22.000	700	33.000
1990	25.000	24.000	1.000	50.000
1991	37.000	31.500	1.500	70.000
1992	48.000	38.000	2.000	88.000

Andamento tendenziale del mercato Page Printer
(Per gentile concessione Epson Italia)

sizionamenti reciproci delle varie parti, risulteranno rigorosamente identici.

Nell'apposito riquadro, a dimostrazione di quanto asserito, sono presenti quattro rettangoli contenenti grafica e testo che, all'origine, risultavano rigorosamente identici e generati con il programma di impaginazione Ventura (che consente piccoli interventi geometrici).

Il lettore può notare da solo l'enorme differenza riscontrabile tra una stampante ad aghi dotata, al momento della prova, di nastro inchiostrato relativamente nuovo (inutile indicare quale è tra le quattro...) e gestita dal pacchetto **Freedom of Press**; un altro riquadro (e anche stavolta non vi diciamo quale) è stato realizzato con il sistema abitualmente usato dalla Systems Editoriale per stampare la nostra rivista (unità di fotocomposizione capace di 1200 x 1200 dot). Gli altri due riquadri sono stati realizzati con due stampanti laser diverse; la prima di queste, dotata di gestione originale PostScript, non presenta sostanziali differenze dall'altra (non PostScript, ma gestita da un computer su cui girava Freedom of Press). I tempi di stampa, però, variano in maniera considerevole a seconda del sistema usato.

PostScript ad aghi?

Con una stampante ad aghi, quindi, è possibile avere a disposizione la "tecnologia" PostScript; lo dimostra il pacchetto applicativo Freedom of Press.

Basta inserire opportunamente una Rom contenete il "sistema" ed un paio di mega di Ram; operazioni certamente non possibili ad un hobbista, ma certamente realizzabili in una qualsiasi fabbrica di printer.

La possibilità di acquistare stampanti di questo tipo, a nostro modesto parere, contribuirebbe ad un rinnovo del parco macchine "intermedio", in grado di soddisfare le esigenze di quella fascia di utenti che, non disponibile a spendere **sei - sette milioni** per entrare in possesso di una laser PostScript, spenderebbe volentieri un milione e mezzo per una 24 aghi PostScript. Molti di questi potenziali utenti (appartenenti alla fascia medio alta) potrebbero anche decidersi al successivo acquisto di una laser, ovviamente PostScript. Il mercato di queste ultime, poi, non verrebbe certamente compromesso dall'introduzione di modelli ad aghi più economici, inevitabilmente più lenti e con risoluzione meno definita (ma nettamente migliori, come tempi di elaborazione, rispetto a comuni macchine ad aghi bit - mapped).

Rimane da chiedersi quindi, prima di concludere, perché mai nessuno ha pensato, finora, a prendere accordi con la Adobe per la realizzazione di tali stampanti.

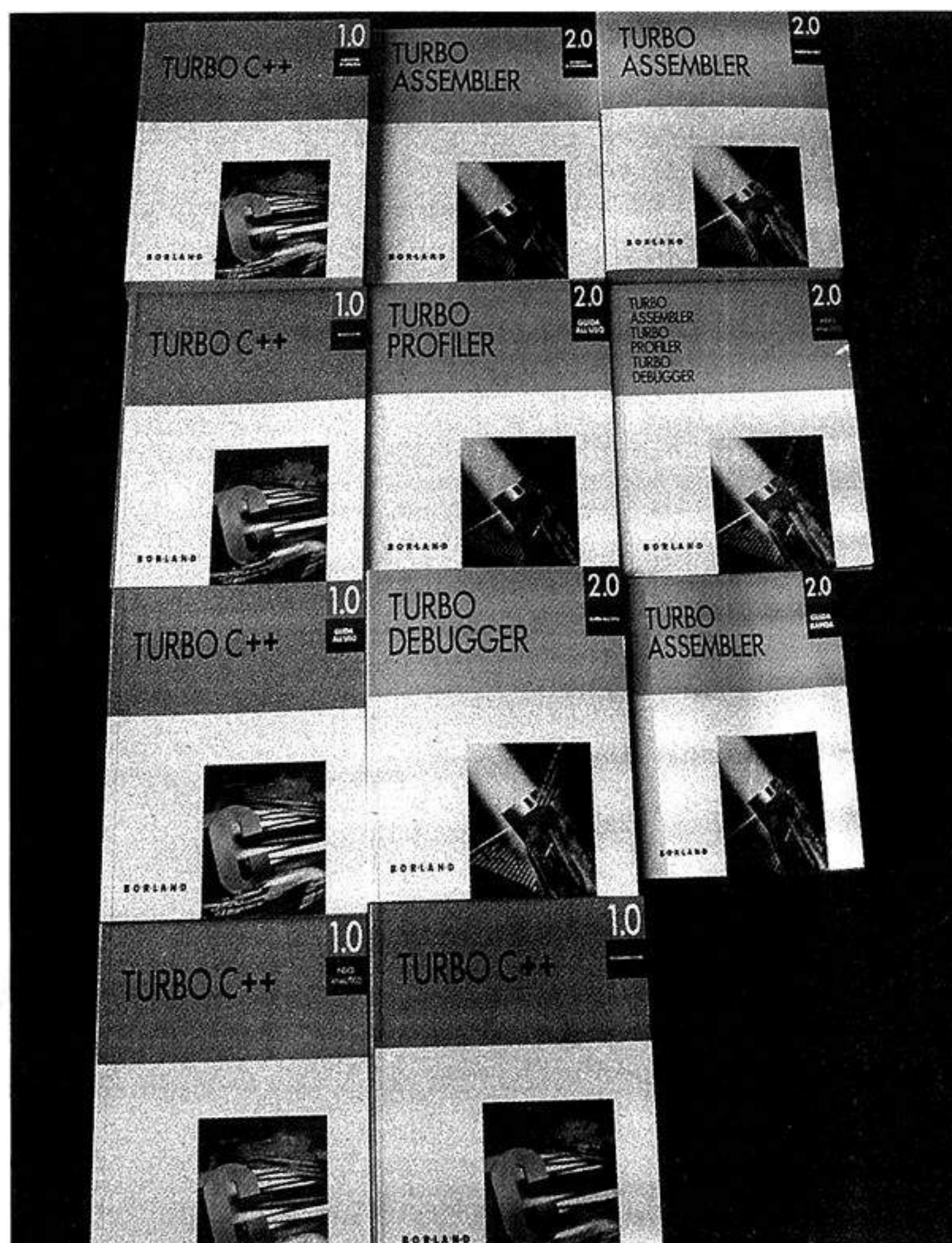
Certo è che se la Adobe pretendesse tre milioni come diritti di licenza su ogni stampante ad aghi prodotta (dal valore intrinseco di mezzo milione) nessuno comprerebbe simili periferiche!

Se, però, i (giusti) diritti di autore venissero ridimensionati, forse gli utenti finali (cioè noi), i fabbricanti di stampanti e la stessa Adobe (cioè, tutti) potrebbero trarre diversi vantaggi...

Andamento tendenziale del mercato Italiano stampanti
(Per gentile concessione Epson Italia)

di Giancarlo Mariani

Super Turbo C targato Borland

Un compilatore è valido se la velocità di elaborazione dei programmi generati è paragonabile a quella del linguaggio Assembly; il prodotto Borland, infatti...

Ed ecco che la Borland ci ha riprovato. Dopo aver indiscutibilmente dominato il mercato per parecchio tempo con Turbo C 2.0, oggi presenta il nuovissimo **Turbo C++** 1.0, che amplia tutte le caratteristiche della precedente versione con innovazioni veramente straordinarie.

Anche la nuova versione del compilatore propone un ambiente di programmazione integrato. Per le (poche) persone che ancora non sanno che cosa si intenda per **ambiente integrato**, possiamo dire che rappresenta l'unione di diversi programmi idonei a sviluppare software senza passare continuamente da un programma all'altro.

Per capire meglio, facciamo un esempio. Se ci si riferisce ai compilatori di qualche anno fa, un programmatore che avesse voluto sviluppare un programma con tali compilatori avrebbe dovuto svolgere le seguenti operazioni:

1- Digitare il listato sorgente, tramite un *Editor.*

2- Salvare il listato sorgente e quindi uscire dall'editor.

3- Compilare il programma tramite il *Compilatore.*

4- Prendere nota degli (inevitabili) errori; quindi riprendere dal passo 1 per correggere gli errori nel listato sorgente.

5- Una volta eliminati gli errori di sintassi, controllare gli errori logici tramite un programma chiamato *Debugger.*

6- Apportare le correzioni agli errori logici, riprendendo dal passo 1.

7- A questo punto, eliminati gli errori, bisognava collegare al programma tutte le librerie ed i files necessari al suo fun-

zionamento, tramite un programma denominato *Linker*.

8- A questo punto (finalmente!) si poteva mandare in esecuzione il programma sviluppato.

Tutta la sequenza di operazioni descritta, pur possedendo un computer estremamente veloce, comportava un notevole dispendio di tempo, ed anche una certa noia, dovendo passare continuamente da un programma all'altro.

Un ambiente di programmazione integrato comprende, invece, tutte le funzioni viste nei punti sopracitati e passa da una funzione all'altra semplicemente tramite la pressione di uno o più tasti. Inoltre tutte le funzioni sono collegate tra di loro dal momento che si può passare tranquillamente dall'editing del programma sorgente, alla compilazione, al debug, all'esecuzione, senza dover continuamente caricare programmi.

Le funzioni descritte vengono svolte dal Turbo C++ in maniera davvero superlativa; se, poi, si considera l'utilizzo di menu a tendina, la presenza di numerosissime finestre, gli help in linea, numerosi parametri completamente personalizzabili, la possibilità di usare la memoria estesa, e tante altre utilissime caratteristiche, si capisce come il **Turbo C++** costituisca un indispensabile strumento di programmazione per coloro che vogliono sviluppare progetti anche lunghi e complessi.

Inoltre, dato che le funzioni di compilazione, linking (e tutte le altre) sono richiamate automaticamente, il programmatore può interamente dedicare il suo tempo alla stesura del programma, senza preoccuparsi di richiamare editor, linker, librerie, o attivare altre funzioni.

Il Turbo C++ si presenta, nella versione con manuali in italiano, in una confezione piuttosto spaziosa comprendente **8** dischetti e **4** manuali (per un totale di più di **1500 pagine**) oltre alla licenza d'uso ed altri documenti di informazione generale. Il compilatore richiede, per funzionare, un computer **Ms - Dos / Ibm / Pc / Xt / At / Ps2** oppure compatibile, il sistema operativo Ms - Dos **2.0** (o più recente), almeno **640 k** di Ram, un monitor ad **80 colonne** (non importa se monocromatico oppure a colori), un'unità a minidischi (**3.5"** oppure **5 1/4"**) e, cosa indispensabile, un **Hard Disk.**

Prima di iniziare a lavorare, il Turbo C++ va installato sul disco fisso. Per fare ciò si ricorre al programma **Install** (memorizzato in uno dei dischetti) che provvede a leggere il contenuto degli 8 floppy (a proposito, sono disponibili dischetti nei due formati classici), a decompattarlo, unirlo, ed a trasferirlo su Hard Disk. Install chiede poche informazioni all'utente prima di lavorare in completa autonomia, creando su disco fisso le directory necessarie e trasferendovi i files opportuni.

Una volta installato, il compilatore si può personalizzare a piacere, scegliendo ad esempio il tipo di **scheda grafica** presente nel computer, il tipo di **processore** (086, 286, 386), l'eventuale presenza di un **coprocessore** matematico, i tasti funzionali dell'editor, vari parametri di compilazione, e tutto ciò, insomma, che può essere utile per lo sfruttamento ottimale dell'Hardware.

In ogni caso il compilatore riconosce automaticamente tutti i parametri e si "adatta" per funzionare correttamente sul computer sul quale è stato installato; non è quindi obbligatorio far girare il programma di personalizzazione dopo aver installato il pacchetto.

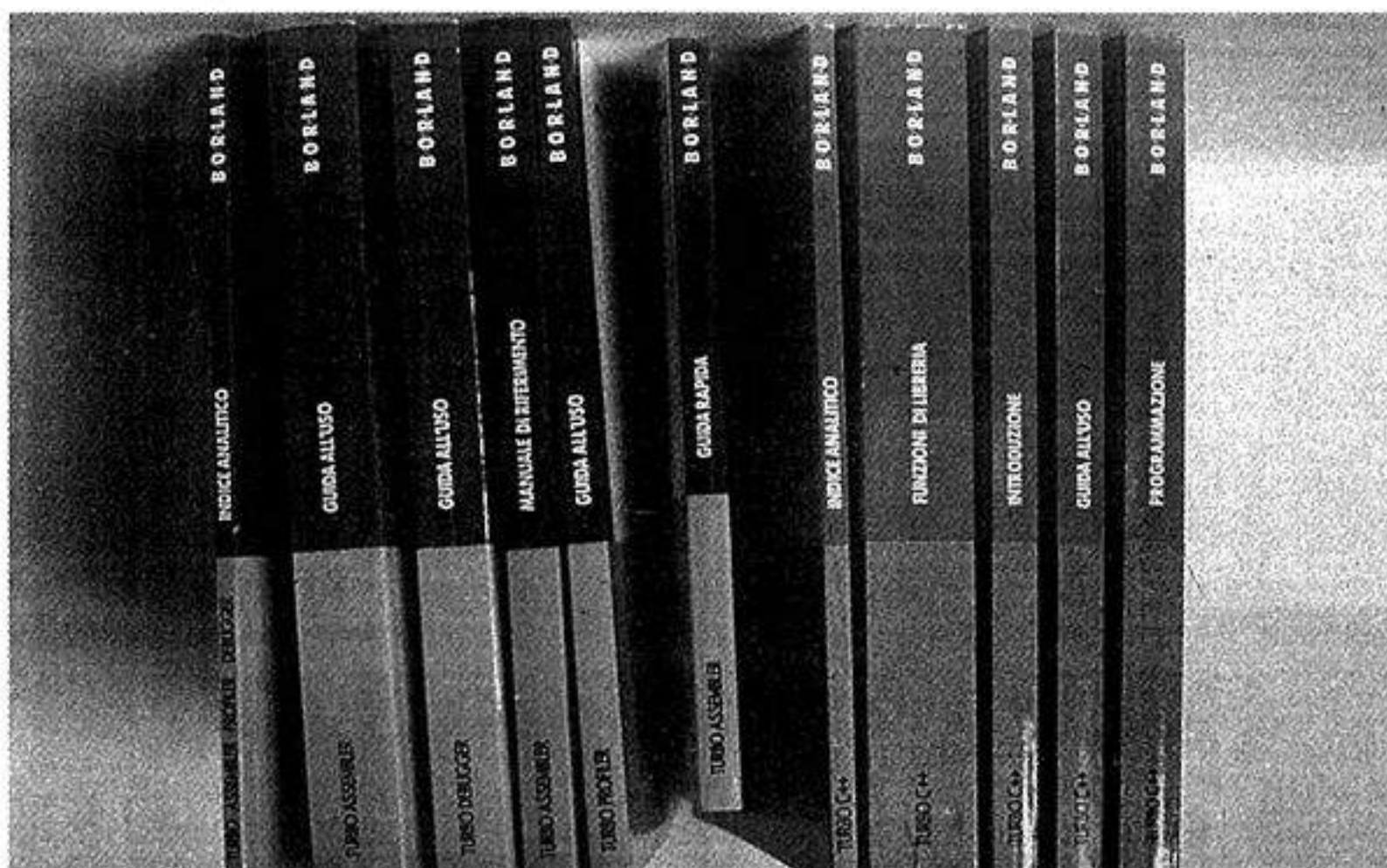
Turbo C++

Descrivere, anche sommariamente, un linguaggio di programmazione nelle poche pagine tradizionalmente riservate ad una recensione è praticamente impossibile. Ci limiteremo, pertanto, a descriverne le caratteristiche più salienti, rimandando, chi desidera maggiori informazioni, a rivolgersi direttamente a chi lo produce, ossia la Borland.

Innanzitutto bisogna dire che un compilatore è un programma in grado di convertire un file sorgente, scritto in un linguaggio ad alto livello (in questo caso, **C**), in un file oggetto in grado di essere eseguito direttamente da Dos senza che sia necessaria la presenza del compilatore stesso.

Nel caso del C, la traduzione viene effettuata in un linguaggio molto vicino al linguaggio macchina del processore, ottenendo così programmi estremamente veloci, almeno una decina di volte più veloci dei corrispondenti programmi scritti in linguaggi più lenti, come il Basic.

Turbo C++, come detto prima, è un ambiente integrato, e tutte le sue funzioni si possono richiamare tramite **menu a tendina**, attivabili con la pressione di un tasto. Le funzioni disponibili sono troppe per elencarle tutte; bisogna solo dire che usando l'ambiente integrato è praticamente impossibile commettere errori. Infatti, le opzioni dei menu non permesse in un determinato momento saranno "spente", rendendo impossibile la loro selezione.

In questo modo non si potrà far partire un programma che contiene errori, nè

1500 pagine di manualistica non sono uno scherzo...

linkare files inesistenti od altre amenità del genere.

Un'altra interessantissima caratteristica del pacchetto è quella di avere un **Help** sempre in linea, gestito tramite **ipertesto**. Vediamo come funziona:

Se state scrivendo un programma, e non ricordate come e quando si può usare una certa **parola chiave** o una **variabile riservata**, è sufficiente posizionarsi sulla parola che solleva il dubbio e premere il tasto di aiuto. Immediatamente apparirà una finestra contenente una spiegazione abbastanza approfondita della parola in oggetto.

Ma non solo! In quasi tutti i casi è presente anche un esempio di programmazione che mostra un uso pratico del comando. Inoltre, tramite le funzioni *taglia e incolla* dell'editor, sarà possibile estrarre qualsiasi esempio (o parte di esso) dalla finestra di aiuto e riportarla sul listato sorgente. In questo modo, potremo utilizzare tutti gli esempi di programmazione ed adattarli facilmente ai nostri programmi.

Altra interessante caratteristica dell'ipertesto è che nella finestra di help relativa al comando prescelto è possibile trovare parole o definizioni che hanno una qualche attinenza con il comando in oggetto, e quindi richiedere l'help anche di queste ultime semplicemente posizionandosi con il cursore su di esse e premendo il tasto di aiuto.

Una caratteristica nuova, per pacchetti integrati di questo genere, è che se si richiede l'help di una parola che non esiste, Turbo C farà apparire una finestra con l'indice alfabetico di tutte le "cose" delle quali si può richiedere l'help e si posizionerà automaticamente sulla parola che assomiglia di più a quella per la quale si è richiesto l'aiuto.

Tale particolare permette di determinare eventuali errori di sintassi (o di digitazione) prima di compilare il programma.

Naturalmente non poteva mancare, in un pacchetto del genere, un **Debugger Integrato**, ossia uno strumento che permette di controllare la corretta esecuzione dei programmi.

Tramite il Debugger è possibile inserire **Breakpoints** o **Watchpoints** nel programma, visualizzare variabili e loro valori durante l'esecuzione, eseguire il programma passo-passo, e moltissime altre caratteristiche, che ne fanno uno strumento utilissimo per l'individuazione di errori logici, che altrimenti richiederebbero un notevole dispendio di tempo.

L'editor integrato nel pacchetto assomiglia (badate bene: *assomiglia* soltanto) a quello delle precedenti versioni di **C** o di **T. Pascal**. Di fatto è pienamente compatibile con questi, ma sono state aggiunte numerose funzioni che evitano le (piuttosto) macchinose operazioni delle versioni precedenti.

Ad esempio, per marcare un blocco, l'editor usava comandi del tutto identici al famoso word processor **WordStar**, ossia **Ctrl-K-B** (per marcare l'inizio), **Ctrl-K-K** (per indicarne la fine), **Ctrl-K- S** (per salvare il blocco), e così via.

Nella nuova versione tutte queste operazioni sono ancora disponibili, ma è anche possibile marcare blocchi semplicemente mantenendo premuto il tasto Shift e selezionando, tramite cursore, il blocco desiderato.

Una volta selezionato, lo si potrà inserire nella memoria con i tasti **Ctrl-Ins**, oppure cancellarlo tramite **Del**, o ancora copiarlo tramite **Shift-Ins**, ricalcando il modo di operare in pacchetti del tipo **QuickBasic**.

L'**editor** del Turbo C++ gestisce realmente moltissimi files contemporaneamente, ognuno in una diversa finestra. Le varie finestre, comprese quelle speciali di watch, lo schermo utente, i messaggi del compilatore, ed altro si possono selezionare tramite la pressione di un tasto.

La gestione delle finestre nel Turbo C è molto complessa ed articolata: si possono avere finestre con i vari files sorgenti ed i messaggi, una sopra l'altra, una di fianco all'altra, una dentro l'altra.

Tali possibilità permettono di editare i vari sorgenti in modo semplice, guardando solo quelli che interessano e spostandosi con estrema facilità da un sorgente all'altro. Inoltre il pacchetto può essere settato per il salvataggio automatico dei sorgenti modificati, permettendo così di rimediare ad alcune "dimenticanze" dei programmatori che frequentemente perdono modifiche e (soprattutto) tempo.

I manuali

I manuali, lo ripetiamo per l'ennesima volta, sono **indispensabili** in un pacchetto del genere, dal momento che non ci troviamo di fronte ad un programmino per Vic 20 (per il quale, nella maggior parte dei casi, una decina di fotocopie erano più che sufficienti).

Nel caso di un pacchetto come il Turbo C, la complessità ed il numero delle funzioni presenti (1500 pagine di manuali ne sono la prova) dimostrano l'alta professionalità del prodotto. Tutt'al più, senza manuali, si può "giocare" con il pacchetto, limitandosi a sviluppando *semplici* programmi di prova, precludendosi però la totalità delle potenti caratteristiche che lo caratterizzano.

D'altronde, un generico manuale sul C non è affatto sufficiente per usare bene il

pacchetto; non c'è, infatti, solo il linguaggio da capire, ma soprattutto il corretto *uso* del compilatore, che permette di sfruttarne al meglio le potenzialità. Bisogna inoltre pensare che l'acquisto della **confezione originale** porta ad altri (non indifferenti) **vantaggi**, tra i quali la possibilità di acquistare la prossima versione del prodotto (pagando una minima integrazione), il diritto a consulenze telefoniche gratuite per qualsiasi problema potesse insorgere sull'uso del pacchetto, informazioni su novità e prodotti della casa. I manuali di cui ci occupiamo si riferiscono alla versione italiana del compilatore Turbo C++ 1.0:

Introduzione

(**260** pagine) Contiene spiegazioni su come creare, ed eseguire, programmi in Turbo C (cioè su come usare il pacchetto nel più breve tempo possibile).

Il primo capitolo spiega come installare il Turbo C++ sul sistema; il secondo capitolo introduce alcune delle più interessanti caratteristiche del linguaggio; il terzo capitolo tratta dell'ambiente integrato, dell'editor, dell'uso del mouse: insomma, una panoramica sul prodotto; il quarto capitolo è una introduzione al linguaggio C, che inizia dalle prime caratteristiche del linguaggio; il quinto capitolo è una esercitazione in C riguardo alla programmazione orientata agli oggetti; il sesto capitolo è una introduzione pratica al C++; il settimo capitolo descrive il Debugger integrato di Turbo C++ e ne presenta, attraverso validi esempi, le numerose caratteristiche.

Guida all'uso

(**250** pagine) E' un approfondimento sulle varie caratteristiche del linguaggio, sull'ambiente integrato, sul nuovo (rispetto alla versione precedente) gestore di progetti, sul nuovo editor, sui vari programmi di servizio.

Nel capitolo 1 è presente la descrizione completa dell'ambiente di sviluppo; il capitolo 2 spiega come creare e gestire progetti composti da più files sorgenti; il capitolo 3 descrive dettagliatamente il nuovo editor; il capitolo 4 spiega come usare il compilatore a riga di comando TCC da Dos e come configurarlo a seconda delle proprie esigenze; il capitolo 5 descrive alcuni dei programmi di servizio inclusi nel pacchetto Turbo C++.

In appendice è spiegato come cambiare e personalizzare le caratteristiche dell'editor integrato, usando un linguaggio appositamente studiato.

Programmazione

(**380** pagine) Il volume contiene materiale utile per i programmatori esperti di C e C++, con la descrizione di funzioni speciali.

Il capitolo 1 descrive il linguaggio Turbo C++, riportando le estensioni rispetto allo standard ANSI-C con eventuali confronti tra istruzioni simili; il capitolo 2 fornisce utili informazioni sul codice sorgente della libreria run-time e descrive i vari files di intestazioni (suffisso **.H**) del C; il capitolo 3 spiega come usare la libreria di Stream del C++; il capitolo 4 si occupa di tutto ciò che concerne i modelli di memoria e gli Overlay; il capitolo 5 è dedicato al video, sia in modo testo che grafico; il capitolo 6 spiega come interfacciare i programmi Turbo - C con i programmi Assembler; il capitolo 7 elenca i messaggi di errore generati durante la compilazione (e l'esecuzione) del programma e suggerisce eventuali soluzioni.

In appendice è spiegato come turbo-c opera rispetto ad alcuni aspetti dello standard Ansi-C.

Funzioni di libreria

(**570** pagine) Questo manuale contiene una lista completa e dettagliata di tutte le istruzioni del Turbo-C++, nonché delle variabili globali.

Il capitolo 1 è una descrizione, ordinata alfabeticamente, di tutte le funzioni di libreria, descrivendo la sintassi, i files di Include, eventuali valori restituiti, un esempio di programmazione ed un'annotazione sulla "portabilità" su altri compilatori C; il capitolo 2 tratta delle variabili globali del Turbo - C++.

❑ ❒

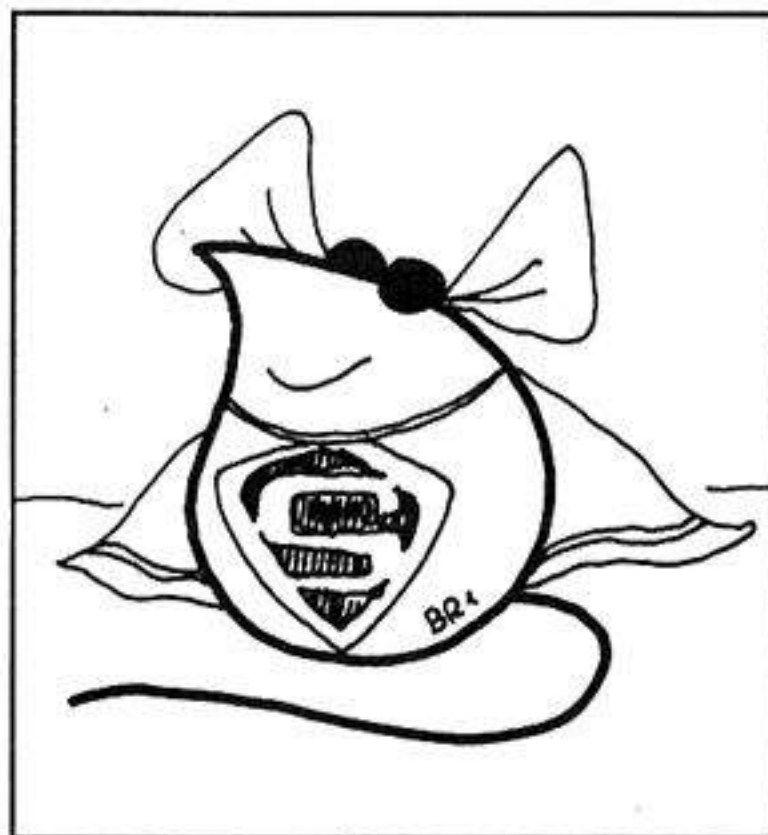
Gli "Oggetti"

Turbo - C++ è un linguaggio "Object Oriented", ossia *orientato agli oggetti*. Spiegare di che cosa si tratta in poche righe è praticamente impossibile; sappiate, comunque, che la programmazione orientata agli oggetti è una interessante caratteristica riservata a linguaggi di altissimo livello, dal momento che consente di creare **moduli di programmi** idonei, a loro volta, a richiamare altri moduli senza sapere come questi ultimi sono stati realizzati; consente, in altre parole, di risparmiare molto tempo nelle modifiche al software che si sta scrivendo e di modificare programmi (scritti in precedenza) conoscendone solo poche, essenziali caratteristiche.

Tale descrizione farà probabilmente sorridere i conoscitori di OOP, ma in poche righe non si può proprio fare di più.

Il Turbo - C++ possiede una particolare libreria (Classlib) interamente dedicata alla programmazione orientata agli oggetti, ma il suo uso è dedicato ai super-esperti, già conoscitori del compilatore.

Turbo - C++, per chi non lo ha ancora capito(!), rappresenta una meravigliosa simbiosi di numerose potenzialità, tutte richiamabili tramite menu a tendina e interconnesse l'una con l'altra. Il pacchetto può rappresentare un validissimo strumento di programmazione non solo per le software-house, ma, grazie al prezzo ridott(issim)o, anche per i programmatori dilettanti e per gli hobbisti (ovviamente evoluti) che vogliono approfondire la conoscenza del linguaggio C, uno dei più efficienti e "vicini" all'hardware delle macchine basate su processori della serie **80X86**.

Per maggiori informazioni:
Borland Italia
Via Cavalcanti, 5
20127 Milano
Tel. 02/26.10.10.2

by Giancarlo Mariani

Le istruzioni Cmp, Inc, In, Jc, Jn

Aumentare, decrementare, confrontare... come è dura la vita del processore 80X86!

La volta scorsa abbiamo visto come eseguire **salti condizionati** a seconda del valore di un flag, precisamente del flag di zero (**ZF**).

Le istruzioni erano **JZ** (salta se ZF = 1) e **JNZ** (salta se ZF = 0). Abbiamo visto anche un modo per condizionare il flag ZF, ossia l'istruzione **DEC** che decrementa un registro (o il contenuto di una locazione di memoria) e, a seconda del valore di quest'ultimo, setta o resetta il

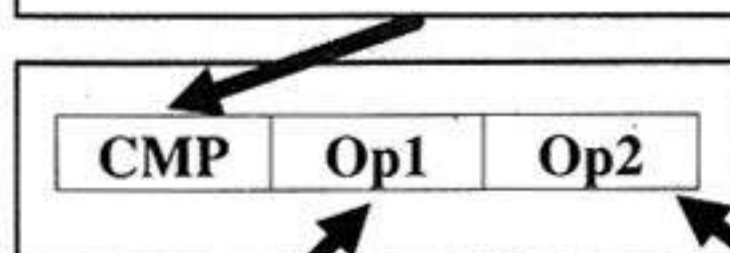
Alcuni esempi di istruzione CMP:

```
CMP  BX, CX         ; Confronto tra i registri BX e CX
CMP  AL, 57         ; Registro - Valore immediato
CMP  CX, Word1      ; Registro - Locazione di memoria
CMP  Loc1, Loc2     ; Confronto tra locazioni di memoria
CMP  [BP], SI       ; Tabella puntata da BP- SI
CMP  DX, 1234h      ; Confronta BX con il valore 1234h
```

CMP

CMP è il codice mnemonico dell'istruzione, abbreviazione dell'inglese CoMPare, ossia confronta.

L'istruzione confronta Op1 con Op2 quindi:
Se Op1 = Op2 setta il flag di zero (ZF) = 1.
Se Op1 < Op2 il flag carry (CF) è messo a 0.
Se Op1 >= Op2 il flag carry (CF) è messo a 1.
Quindi dopo CMP si può controllare il flusso del programma tramite le note istruzioni di salto condizionato.

Op2: E' il secondo operando, che può essere specificato da:
Un qualsiasi registro del processore (AX, BX, CX...) anche nella sua forma ad 8 bit (AL, BL, CL...) tranne i registri speciali (Flags e IP, Instruction Pointer).
Una locazione di memoria, anche specificata da una label.
Un valore immediato

Op1: E' il primo operando, che può essere specificato da:
Un qualsiasi registro del processore (AX, BX, CX...) anche nella sua forma ad 8 bit (AL, BL, CL...) tranne i registri speciali (Flags e IP, Instruction Pointer).
Una locazione di memoria, anche specificata da una label.
Una tabella in memoria indirizzata dal registro BX o BP.

Attenzione a:

I due operandi **Op1** e **Op2** devono sempre essere dello stesso tipo, ossia bytes **oppure** word. Non sarà quindi possibile eseguire confronti del tipo...

```
CMP AX, BL
```

...perché Op1 (AX) è lungo **16** bit, mentre OP2 (BL) solo **8**. Sarà invece possibile effettuare...

```
CMP AX, BX
```

...oppure...

```
CMP AL, BL
```

L'istruzione **CMP AX, Loc1** confronta AX (16 bit) con le locazioni di memoria "Loc1" e "Loc1 + 1", mentre l'istruzione **CMP AL, Loc1** confronta AL (8 bit) con la sola locazione di memoria Loc1. Come si vede il modo di scrivere Op2 non cambia, ma questo verrà considerato lungo come Op1.

CMP influenza i flag AF, CF, OF, PF, SF, ZF.

flag ZF permettendo così un controllo con le istruzioni di salto condizionato.

Il sistema descritto, tuttavia, non permette di controllare se un certo *registro* o una *cella* di memoria sia arrivata ad un valore predeterminato, ma solo se è arrivata a 0. In pratica, un'istruzione del tipo Basic...

```
IF ax = 156 THEN GOTO Label1
```

...non sarà mai possibile, o sarà estremamente complicata, utilizzando la sola istruzione Dec.

Per ottenere lo scopo prefisso ci serviremo, invece, di un'altra istruzione Assembler, ossia **CMP** (comparazione). Essa permette di comparare un registro (oppure il contenuto di una locazione di memoria) con un altro registro (oppure locazione di memoria o valore immediato) e quindi il flag ZF viene settato opportunamente a seconda del risultato del confronto. La riga Basic vista prima, quindi, in Assembler diventerebbe...

```
CMP ax, 156
JZ Label1
```

Dal momento che CMP influenza alcuni flags, tra i quali quello di Zero (ZF) ed il flag Carry (CF), vedremo come utilizzarla, praticamente, per nostri scopi, ricordando che le due istruzioni di salto condizionato, che controllano il flag di zero, sono già state descritte in precedenza (**JZ** e **JNZ**). Nella scorsa puntata avevamo visto come, con l'istruzione DEC, era possibile decrementare un registro o una locazione di memoria. Stavolta, nel riquadro specifico, vediamo l'istruzione INC, che serve per incrementare.

L'istruzione Int

L'ultima istruzione della quale ci occupiamo in questa puntata è **INT** (da INTerrupt).

Questa è un'istruzione molto particolare, che serve per richiamare routines posizionate nel **Bios**, nel **Dos** oppure delle routines costruite dal programmatore per svolgere determinati compiti. Le funzioni di cui ci occuperemo stavolta sono dedicate all'input / output di **caratteri sullo schermo**, appunto già presenti nel Bios e nel Dos del PC e, come tali, pronte per

JC / JNC

Destinazione. E' l'indirizzo, o meglio, l'offset da aggiungere all'attuale contatore di programma per effettuare il salto. Questo è un salto relativo, ossia può variare tra -128 e +127 bytes dal punto di partenza. Generalmente l'indirizzo di destinazione nelle istruzioni di salto relativo si specifica con un'etichetta.

Tutti gli indirizzi specificati nelle istruzioni JC / JNC fanno ovviamente riferimento al code segment CS.

JC JNC	Destinazione (Indirizzo tipo Short)

Alcuni esempi di istruzione JC / JNC:

```
JC   Label1        ; Salta a Label1 se CF = 1
JNC  Pippo         ; Salta a Pippo se CF = 0
```

JC / JNC (Jmp on Carry / Jmp on Not Carry) sono i codici mnemonici delle due istruzioni.

JC salta se il flag di carry CF è uguale a 1.

JNC esegue l'esatto contrario, ossia salta se CF è uguale a zero.

Tenendo conto di quanto detto per l'istruzione CMP (a proposito di ZF e CF), si possono fare alcuni **esempi**:

```
CMP AL, 20
JZ  Label1    ;Vai a "Label1" se AL = 20
CMP AX, BX
JC   Maggiore ;Vai a "Maggiore" se AX >= BX
CMP Loc1, 35h
JNZ Diversi ;Vai a "Diversi" se Loc1 <> 35h
CMP DX, 1000
JNC Minore  ;Vai a "Minore" se DX < 1000
```

Attenzione a:

La parte di programma alla quale si salta tramite le istruzione JC / JNC deve contenere corrette istruzioni Assembler 80X86, perché se si salta ad una parte di memoria non contenente un programma, il computer si inchioda.

Le istruzioni non influenzano alcun FLAG.

essere utilizzate. INT è un'istruzione semplicissima...
`INT Interrupt`
...in cui **Int** è il codice mnemonico dell'istruzione (abbreviazione dell'inglese INTerrupt, ossia interruzione) ed **Interrupt** è il **numero** dell'interruzione da richiamare, e può essere **SOLO** un valore immediato. Si sottolinea che ad ogni Interrupt sono associate numerose funzioni. In particolare, l'Interrupt numero **10h** è contenuto nelle Rom del Bios, mentre l'Interrupt **21h** è presente nel Dos. Ci limiteremo ad occuparci di alcune **funzioni** di tali Interrupt. Cominciamo con 10h. Attivando l'istruzione Assembler `INT 10h`, si richiama l'Interrupt Bios numero 10h (**16** decimale) tramite cui è possibile svolgere numerosissimi compiti, principalmente dedicati alla gestione del video. Ovviamente non sarà sufficiente richiamare l'Interrupt per eseguire una qualsiasi funzione, ma saranno necessari altri parametri per definire la funzione; in particolare:

Il **registro ah** deve essere definito prima del richiamo dell'Interrupt 10h, e deve contenere la funzione che questo deve svolgere. Gli altri registri servono come parametri di ingresso / uscita. Le funzioni dell'Interrupt 10h, delle quali ci vogliamo occupare, sono:

INT 10h, Funzione 07h

(Inizializza la finestra video); viene utilizzata per **cancellare** lo schermo o parte di esso. I parametri da passare alla funzione sono:

ah = 07h (Funzione 7)
al = 0
bh = Colore da usare per riempire l'area cancellata.
ch = Coordinata Y dell'angolo in alto a sinistra

INC

Destinazione. E' il parametro che va incrementato, al quale va aggiunto il valore unitario.
Destinazione può essere specificato da:
Un qualsiasi registro del processore (AX, BX, CX...) anche nella sua forma ad 8 bit (AL, BL, CL...) tranne i registri speciali (Flags e IP, instruction pointer).
Una locazione di memoria, anche specificata da una label.
Una tabella in memoria indirizzata dal registro SI.
I dati trattati dall'istruzione fanno sempre riferimento al Data Segment DS.

INC è il codice mnemonico (Assembler) dell'istruzione, ed è l'abbreviazione dell'inglese "INCrement", ossia incrementa. Come facilmente intuibile, l'istruzione serve per incrementare (addizionare 1) il parametro "Destinazione".

Seg (facoltativo) specifica il segmento contenente i dati da trattare, e può essere SS, DS, ES, CS. Nel caso di trattamento di dati da un altro segmento, infatti, bisogna specificarlo **prima** del parametro destinazione, con il nome del registro di segmento desiderato seguito dal carattere doppio punto (:)

Alcuni **esempi** di istruzione INC:

```
INC  CX             ; Registro a 16 bit
INC  SI             ; Registro a 16 bit
INC  DL             ; Registro ad 8 bit
INC  5345h          ; Locazione assoluta di memoria
INC  Contatore      ; Etichetta
INC  Tabella[SI]    ; Tabella in memoria puntata da SI
```

Attenzione a:

In caso di incremento di registro ad 8 bit, DEC occuperà due bytes di memoria; nel caso, invece, di incremento di un registro ad 8 bit, ne occuperà uno solo.

L'incremento di locazioni di memoria è più lento di quello dei registri. Per incrementare una locazione molte volte è meglio caricarne il contenuto in un registro ed agire su questo prima di trasferire il valore finale nuovamente nella locazione di memoria.

L'istruzione INC influenza i flags AF, OF, PF, SF, ZF.

Programma Assembler per cancellare lo schermo di formato 80 x 25

```
mov  ah, 07h          ;Funzione 7
mov  al, 0
mov  bh, 07h          ;Colore (0=Sfondo, 7=Caratteri)
mov  ch, 0            ;Coordinata Y (alto-SX)
mov  cl, 0            ;Coordinata X (alto-SX)
mov  dh, 24           ;Coordinata Y (basso-DX)
mov  dl, 79           ;Coordinata X (basso-DX)
int  10h              ;Esegue la funzione
```

Programma che posiziona il cursore in riga 15 colonna 48

```
mov  ah, 02h          ;Funzione 2
mov  bh, 0
mov  dh, 15           ;Posizione Y = 15
mov  dl, 48           ;Posizione X = 48
int  10h              ;Esegue la funzione
```

Programma Assembler per far apparire una "A" colorata

```
mov  ah, 09h    ;Funzione 9
mov  al, 'A'    ;Carattere "A"
mov  bh, 0
mov  bl, 17h    ;1 = Sfondo Blu, 7 = Carattere bianco
mov  cx, 1      ;Scrive 1 carattere
int  10h        ;Esegue la funzione
```

Programma Assembler per dterminare il tasto premuto

```
mov  ah, 08h          ;Funzione 8
int  21h              ;Esegue la funzione
; al = codice del carattere introdotto.
mov  Caratt, al       ;Salva in "Caratt" il carattere
                      ;introdotto
```

cl = Coor. X dell'angolo in alto a sinistra
dh = Coor. Y angolo in basso a destra
dl = Coor. X angolo in basso a destra

In pratica, per cancellare uno schermo di formato **80** x **25**, il programma Assembler da scrivere sarà del tipo di quello che appare nel riquadro specifico. Questa funzione **NON** posiziona il cursore dopo aver cancellato lo schermo. Bisogna fare attenzione a mantenere le coordinate entro i limiti dello schermo, perché la funzione **NON** le controlla, e potrebbero accadere cose imprevedibili.

INT 10h, Funzione 02h

(Posiziona il cursore); questa funzione è usata per posizionare il cursore all'interno dello schermo, ed è equivalente a **Locate** del Basic. I parametri da passare alla funzione sono:

ah = 02h (Funzione 2)
bh = 0
dh = Riga (Coordinata Y) 0 .. 24
dl = Colonna (Coordinata X) 0 .. 79

Per sistemare il cursore, ad esempio, alla posizione 15, 48 (riga, colonna) un programma Assembler idoneo può essere simile a quello che compare in queste pagine. Se il cursore viene spostato "fuori" dallo schermo (coordinate eccedenti i limiti dello schermo) questo non sarà più visibile.

Int 10h, Funzione 09h :

(Scrive l'attributo ed il carattere sul video); questa funzione viene utilizzata per scrivere sul video, alla posizione corrente del cursore, un carattere con il corrispondente colore. I parametri da passare alla funzione sono:

ah = 09h (Funzione 9)
al = Codice del carattere da scrivere (codice ASCII)
bh = 0
bl = Attributo (colore) del carattere da scrivere (vedi tabella colori).
cx = Numero di volte per cui deve essere scritto il carattere (di solito, 1).

A parte è riportato un programma Assemblerer valido per scrivere una "**A**" alla posizione del cursore con sfondo **Blu** e carattere **Bianco**.

La funzione **NON** incrementa la posiziona del cursore dopo aver scritto il carat-

tere. Deve essere invece aggiornata tramite la funzione 02h.

Nel programma di esempio è stata anche usata una funzione dell'Interrupt 21h, contenuto nel Dos. Vediamola:

INT 21h, Funzione 08h
(Input carattere senza eco); viene utilizzata per inserire un carattere da tastiera, senza produrne l'eco sul video.

Il parametro da passare alla funzione è:

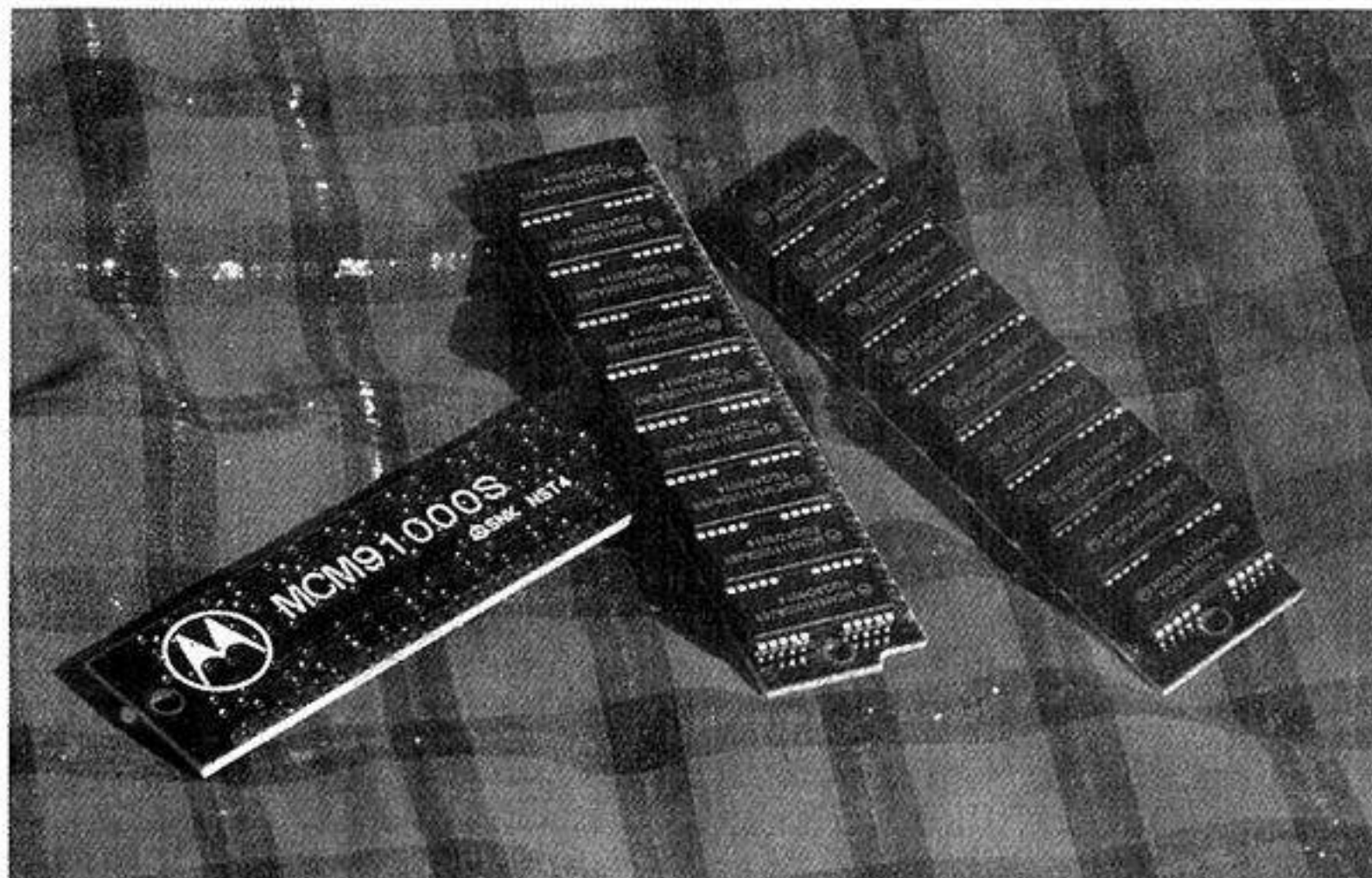
ah = 08h (Funzione 8)

La funzione restituisce in **al** il codice (ASCII) del carattere introdotto da tastiera.

Per leggere un tasto, pertanto, un programma Assembler può essere quello pubblicato a parte.

La funzione blocca l'elaborazione fino a quando non viene introdotto un carattere. Se si premono i tasti Ctrl - C viene prodotto un **Int 23h** ed il programma in esecuzione termina.

Nell'uso dell'istruzione Int bisogna fare moltissima attenzione, innanzitutto a richiamare degli Interrupt *esistenti* ed inoltre a "passare" correttamente i parametri a questi ultimi. Se si richiama un Interrupt non esistente, oppure si sbaglia il passaggio di parametri, il PC potrebbe inchiodarsi, costringendo quindi a resettare.

Una descrizione più completa dei vari Interrupt Dos e Bios (e su come creare proprie routines in Interrupt) verrà data nei prossimi fascicoli. Per ora limitatevi ad usare l'istruzione Int con le funzioni appena descritte.

Ricorderete, a proposito, che anche in precedenza avevamo incontrato l'istruzione Int, ossia:

Int 29h :scrive su video il carattere contenuto in **al**, aggiornando opportunamente la posizione del cursore.

Int 21h :funzione **4Ch**, che interrompe il programma Assembler e consente il ritorno al Dos.

Anche su queste funzioni torneremo dettagliatamente in seguito.

```
;
; CARCOL.ASM: Accetta in input da tastiera dei caratteri e li
;             scrive su video con colori sempre diversi.
;             Termina con CTRL-C
;
cseg        SEGMENT PARA PUBLIC 'CODE'
            org 100h
            ASSUME cs:cseg, ds:cseg, ss:cseg, es:cseg

Start:
            mov  AX,CS
            mov  DS,AX              ;DS = CS
            mov  ES,AX              ;ES = CS

RipetiCiclo:
            mov  Xpos,0             ;Azzera posizione X,Y
            mov  Ypos,0
            mov  Colore,01h         ;Comincia con colore 01h
                                    ;(Sfondo nero, testo blu)
            call Cls                ;Cancella lo schermo
```

Listato Assembler di esempio; prima parte

Il programma di esempio

Il programma di questo mese svolge un compito particolare.

Cancella lo schermo e si pone in attesa della pressione di un tasto; quindi, una volta digitato, lo trascrive fino ad occupare l'intera area del video, incrementando il codice del colore ad ogni nuovo carattere visualizzato.

La posizione del cursore è gestita automaticamente, in modo che vada "a capo" quando raggiunge l'ultima colonna. Se si raggiunge l'ultima riga, lo schermo viene cancellato ed il programma ricomincia dall'inizio.

E' presente anche una gestione del tasto Return, che produce un ritorno carrello (cursore alla prima colonna della riga successiva).

La pressione dei tasti Ctrl - C fa terminare il programma.

Esaminiamo ora il listato.

All'inizio vi sono le "solite" inizializzazioni dei registri di segmento DS ed ES, spiegate nei fascicoli precedenti (a partire dal N. 81 di C.C.C.).

L'istruzione Call Cls chiama, appunto, la subroutine Cls, che serve per cancellare lo schermo.

Il passo successivo richiama l'Interrupt 10h, ed in particolare la funzione 02h, che posiziona il cursore (in questo caso) alle coordinate specificate dalle locazioni Xpos e Ypos.

Una volta posizionato il cursore, tramite l'Interrupt 21h (funzione 8) si aspetta la pressione di un tasto per memorizzarlo nella locazione Char. Premendo CTRL-C il programma termina immediatamente.

L'Interrupt 10h (funzione 9) permette di scrivere il carattere sul video, con il colore di sfondo e testo specificati dalla locazione Colore.

A questo punto la locazione Colore viene incrementata per permettere di scrivere il successivo carattere con un colore diverso; subito dopo si stabilisce (CMP Char, 13) se il carattere introdotto sia un Return o meno. In quest'ultimo caso il programma salta alla label CrKey.

```
AltroChar:
        mov   ah,02h            ;Posiziona il cursore
        mov   bh,0              ;(Pagina video 0)
        mov   dh,Ypos           ;Alle coordinate Ypos.Xpos
        mov   dl,Xpos
        int   10h               ;Richiamo Interrupt Bios

        mov   AH,8              ;Accetta un carattere da tastiera
        int   21h               ;Richiamo Interrupt DOS
        mov   Char,al           ;Salva il carattere in Char

        mov   ah,9              ;Scrive il carattere su video
        mov   bh,0              ;(Pagina video 0)
        mov   bl,Colore         ;Con il colore "Colore"
        mov   cx,1              ;Scrive solo 1 carattere
        int   10h               ;Richiamo Interrupt Bios

        inc   Colore            ;Incrementa colore
        cmp   Char,13           ;Se il carattere è un RETURN (cod.13)
        jz    CrKey             ;Salta a "crkey"

        inc   Xpos              ;Incrementa posizione X
        cmp   Xpos,80           ;Se la posizione X è 80
        jnz   NoColonna80
CrKey:
        mov   Xpos,0            ;Azzera posizione X
        inc   Ypos              ;Incrementa posizione Y
        cmp   Ypos,25           ;Se la posizione Y è 25
        jz    RipetiCiclo       ;Cancella lo schermo e ripete
NoColonna80:
        jmp   short AltroChar   ;Salta a input carattere

;
;Subroutine CLS: Cancella lo schermo
;
Cls:
        mov   ah,07h            ;Cancella lo schermo
        mov   al,0
        mov   bh,07h            ;Colore di sfondo/caratteri
        mov   cx,0              ;Cancella dalla posizione 0.0
        mov   dh,24             ;Alla 24.79
        mov   dl,79
        int   10h               ;Richiamo Interrupt Bios

        ret                     ;Ritorno al programma principale

;
; Dati usati dal programma
;
        Xpos   DB 0             ;Posizione X
        Ypos   DB 0             ;Posizione Y
        Colore DB 0             ;Colore del carattere
        Char   DB 0             ;Carattere introdotto

cseg    ENDS
        END  Start
```

Listato di esempio (continuazione e fine)

Se, invece, non è un Return, viene incrementata la posizione X e se questa raggiunge 80 (fine riga), viene azzerata la posizione X ed incrementata la posizione Y.

Se la posizione Y raggiunge 25 (fine dello schermo) il programma ricomincia dall'etichetta RipetiCiclo.

Dopo aver digitato il programma, lo si può registrare con il nome CARCOL.ASM e quindi lo si può compilare tramite le istruzioni:

MASM CARCOL;
LINK CARCOL;
EXE2BIN CARCOL.EXE CARCOL.COM
DEL CARCOL.EXE

Richiamando il programma da Dos con il comando CARCOL potremo renderci esattamente conto del suo corretto funzionamento. In caso affermativo lo schermo si cancellerà per consentirci di digitare i caratteri e vederli riprodotti, a colori, sullo schermo.

Dal momento che il colore viene semplicemente incrementato ad ogni ciclo, e non vi è alcun controllo su di esso, è possibile che alcuni colori non producano un risultato visibile sullo schermo perché il colore dello sfondo può risultare uguale a quello del carattere.

di Andrea Preziosi

Simulazione grafica del gioco del 15

L'implementazione del notissimo solitario si presta benissimo per approfondire alcune tecniche di programmazione in Gw - Basic

Il programma, che pubblichiamo nella versione per computer **Ms - Dos** compatibili, richiede almeno la scheda grafica **CGA** nella modalità **SCREEN 1**; per quanto riguarda il resto, può girare con qualsiasi configurazione.

Dopo aver dato il Run si cancella lo schermo, vengono disegnate le 15 caselle, circondate da una cornice; di queste, quella in alto a sinistra può essere spostata dai tasti del cursore: premendo il tasto **Return** le caselle si spostano; se, poi, non si riesce a completare il gioco, premendo il tasto **Esc** si può ricominciare una nuova partita o tornare al dos; premendo un tasto sbagliato (oppure indicando una mossa illecita) si sentirà un **Beep**.

Come funziona

I numeri delle caselle sono memorizzati nella matrice *A (4, 4)* che, per esser riempita con numeri da **1** a **15** disposti in maniera casuale, richiede l'uso delle funzioni random (vedi linee da 40 a 90) in cui viene inizializzato il generatore di numeri casuali e aperto un ciclo **Z** da 1 a 15. Una volta determinate le coordinate x e y, se *a (x, y)* vale zero si pone *a (x, y) = z* (e si passa al next); altrimenti vengono trovati due nuovi numeri fino a trovare una casella libera.

Risulta evidente che, trattando 16 caselle con 15 numeri, una casella resta vuota. Fondamentale è dunque l'importanza di conoscere la posizione di quest'ultima per determinare le caselle che si possono spostare e la direzione in cui il movimento è possibile.

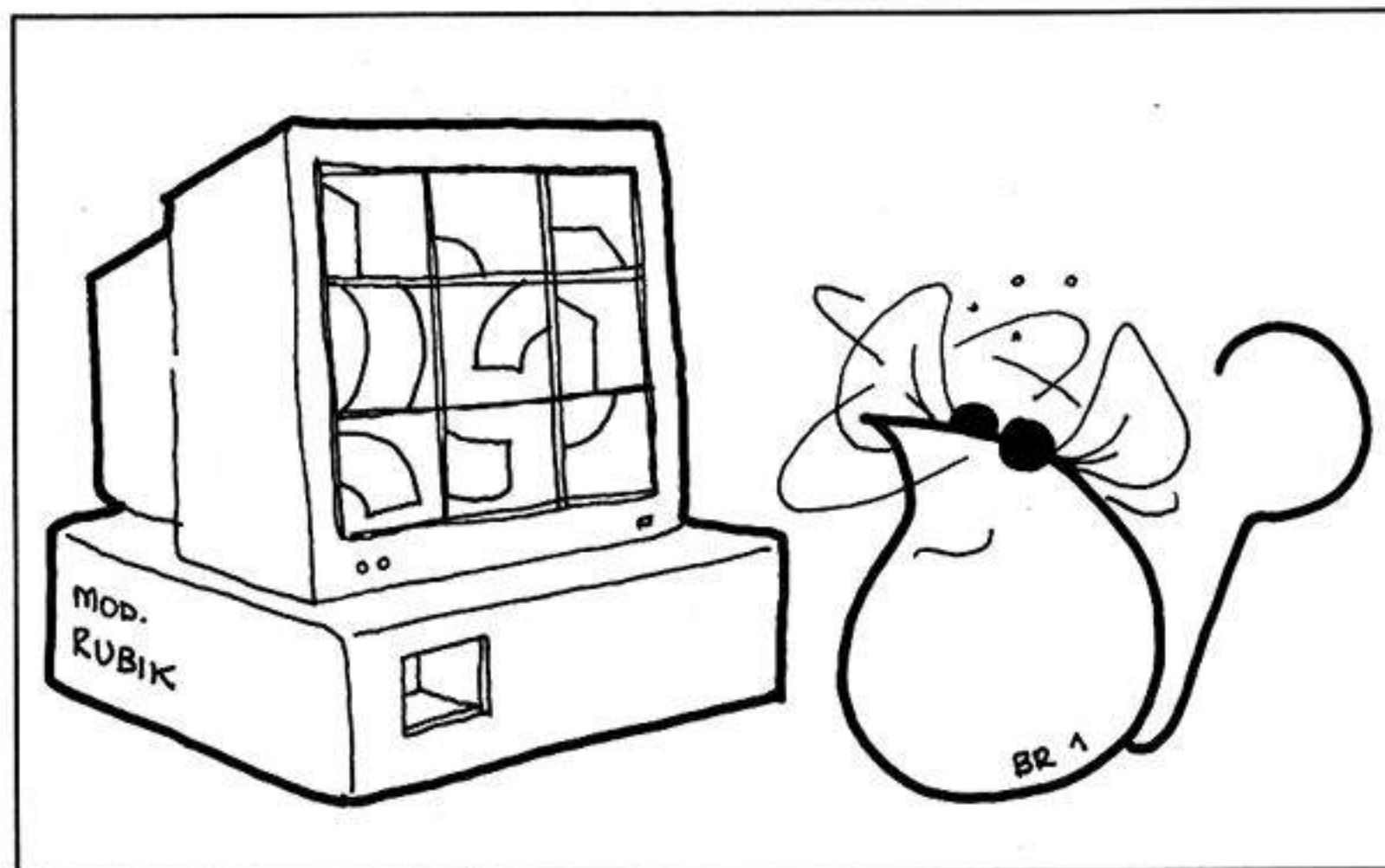
A questo punto vengono aperti due cicli per esaminare gli elementi della matrice, mentre le coordinate della casella vuota vengono conservate nelle variabili **pzx** e **pzy**.

Affinchè una casella possa essere spostata, deve trovarsi nella stessa fila (orizzontale e verticale) in cui è posizionata la casella vuota. Il semplice controllo viene effettuato dalla riga 380, mentre le righe da 400 a 430 servono per determinare da quale parte si devono spostare le caselle sia sullo schermo che nella matrice.

Ecco una descrizione completa delle funzioni svolte dalle linee del programma:

10 - 30 impostazioni iniziali.

40 inizializzazione del generatore di numeri casuali.

50 - 90 riempimento della matrice *a (4, 4)*.

```
1 REM gioco del 15 (by Andrea Preziosi per C.C.C.)
10 CLS : KEY OFF
20 DIM A(4, 4), Z%(385)
30 FOR X = 1 TO 4: FOR Y = 1 TO 4: A(X, Y) = 0: NEXT:
NEXT
40 RANDOMIZE TIMER
50 FOR Z = 1 TO 15
60 X = INT(RND * 4) + 1
70 Y = INT(RND * 4) + 1
80 IF A(X, Y) = 0 THEN A(X, Y) = Z ELSE 60
90 NEXT
100 FOR X = 1 TO 4: FOR Y = 1 TO 4
110 IF A(X, Y) = 0 THEN PZX = X: PZY = Y
120 NEXT: NEXT
130 SCREEN 1
140 FOR X = 1 TO 4: FOR Y = 1 TO 4
150 IF A(X, Y) = 0 THEN 200
160 X1 = 96 + (X - 1) * 32 + 1: X2 = X1 + 29
170 Y1 = 36 + (Y - 1) * 32 + 1: Y2 = Y1 + 29
180 LINE (X1, Y1)-(X2, Y2), 2, B
190 LOCATE INT(Y1 / 8) + 3, INT(X1 / 8) + 2: PRINT USING
"##"; A(X, Y)
200 NEXT: NEXT
210 LINE (94, 34)-(225, 165), 1, B
220 X = 1: Y = 1: M = 0
230 X1 = 96 + (X - 1) * 32: X2 = X1 + 31
240 Y1 = 36 + (Y - 1) * 32: Y2 = Y1 + 31
250 LINE (X1, Y1)-(X2, Y2), 1, B
260 LOCATE 25, 1: PRINT "MOSSE EFFETTUATE:"; M;
270 GOTO 790
280 INK$ = INKEY$: IF INK$ = "" THEN 280
290 IF INK$ = CHR$(27) THEN 720
300 IF INK$ <> CHR$(0) + CHR$(72) AND INK$ <> CHR$(0) +
CHR$(75) AND INK$ <> CHR$(0) + CHR$(77) AND INK$ <>
CHR$(0) + CHR$(80) AND INK$ <> CHR$(13) THEN BEEP: GOTO
280
310 LINE (X1, Y1)-(X2, Y2), 0, B
320 IF INK$ = CHR$(13) THEN 380
330 IF INK$ = CHR$(0) + CHR$(72) THEN Y = Y - 1: IF Y =
0 THEN Y = 4
340 IF INK$ = CHR$(0) + CHR$(80) THEN Y = Y + 1: IF Y =
5 THEN Y = 1
350 IF INK$ = CHR$(0) + CHR$(75) THEN X = X - 1: IF X =
0 THEN X = 4
360 IF INK$ = CHR$(0) + CHR$(77) THEN X = X + 1: IF X =
5 THEN X = 1
370 GOTO 230
380 IF NOT (PZX = X XOR PZY = Y) THEN BEEP: GOTO 230
390 M = M + 1
400 IF X > PZX THEN 440
410 IF PZX > X THEN 510
420 IF Y > PZY THEN 580
430 GOTO 650
440 FOR Z = PZX TO (X - 1): SWAP A(Z, Y), A(Z + 1, Y):
NEXT
```

Gioco del 15 versione Gw - Basic (Ms - Dos). Prima parte

100 - 120 ricerca della casella rimasta vuota.

130 - 210 disegno delle 15 caselle.

220 impostazioni per inizio partita.

230 - 250 disegno della cornice indicante la casella da spostare.

260 stampa del numero di mosse effettuate.

270 salto alla routine di controllo delle caselle.

280 attesa della pressione di un tasto.

290 salto alla linea 720 (se è stato premuto il tasto Esc).

300 input controllato (solo tasti cursore e return).

310 cancellazione della cornice.

320 salto alla riga 380 (se è stato premuto il tasto Return).

330 - 360 spostamento delle coordinate della cornice.

370 salto alla linea che disegna la cornice.

380 controllo della liceità di spostamento.

390 conteggio delle mosse effettuate.

400 - 430 esame della direzione in cui spostare le caselle.

440 - 710 spostamento delle caselle sullo schermo e nella matrice.

720 - 780 menu per inizio di una nuova partita (o uscita).

790 - 930 controllo ordine delle caselle.

940 - 980 si attivano quando si completa il gioco.

Variabili usate

Z cicli for next.

INK$ determinazione dei tasti premuti.

X e **Y** posizione della cornice.

X1, **Y1** e **X2**, **Y2** coordinate grafiche.

M numero di mosse effettuate.

PZX e **PZY** posizione della casella vuota

Spesso ci si può trovare in alcune particolari situazioni che sembrano impedire la conclusione del gioco (per esempio, quando l'ultima fila risulta formata dalle caselle 13, 15, 14). Sarebbe interessante individuare un algoritmo che, esaminando la disposizione casuale iniziale, riesca ad escludere tali situazioni particolari, riattivando la procedura Random.

Numeri casuali

I numeri casuali sono indispensabili per la creazione di giochi o presentazioni.

La funzione **Rnd** restituisce un numero casuale (in semplice precisione) compre-

```
450 LINE (X1, Y1)-(X2, Y2), 0, B
460 X1 = 96 + (PZX) * 32: X2 = 96 + (X - 1) * 32 + 31
470 Y1 = 36 + (Y - 1) * 32: Y2 = Y1 + 31
480 GET (X1, Y1)-(X2, Y2), Z%
490 FOR Z = X1 TO X1 - 32 STEP -1: PUT (Z, Y1), Z%,
PSET: NEXT
500 PZX = X: PZY = Y: GOTO 230
510 FOR Z = PZX TO (X + 1) STEP -1: SWAP A(Z, Y), A(Z -
1, Y): NEXT
520 LINE (X1, Y1)-(X2, Y2), 0, B
530 X1 = 96 + (X - 1) * 32: X2 = 96 + (PZX - 1) * 32 - 1
540 Y1 = 36 + (Y - 1) * 32: Y2 = Y1 + 31
550 GET (X1, Y1)-(X2, Y2), Z%
560 FOR Z = X1 TO X1 + 32: PUT (Z, Y1), Z%, PSET: NEXT
570 PZX = X: PZY = Y: GOTO 230
580 FOR Z = PZY TO (Y - 1): SWAP A(X, Z), A(X, Z + 1):
NEXT
590 LINE (X1, Y1)-(X2, Y2), 0, B
600 X1 = 96 + (X - 1) * 32: X2 = X1 + 31
610 Y1 = 36 + (PZY) * 32: Y2 = 36 + (Y - 1) * 32 + 31
620 GET (X1, Y1)-(X2, Y2), Z%
630 FOR Z = Y1 TO Y1 - 32 STEP -1: PUT (X1, Z), Z%,
PSET: NEXT
640 PZX = X: PZY = Y: GOTO 230
650 FOR Z = PZY TO (Y + 1) STEP -1: SWAP A(X, Z), A(X, Z
- 1): NEXT
660 LINE (X1, Y1)-(X2, Y2), 0, B
670 X1 = 96 + (X - 1) * 32: X2 = X1 + 31
680 Y1 = 36 + (Y - 1) * 32: Y2 = 36 + (PZY - 1) * 32 - 1
690 GET (X1, Y1)-(X2, Y2), Z%
700 FOR Z = Y1 TO Y1 + 32: PUT (X1, Z), Z%, PSET: NEXT
710 PZX = X: PZY = Y: GOTO 230
720 SCREEN 0: WIDTH 80
730 LOCATE 2, 2: PRINT "1) RICOMINCIA"
740 LOCATE 4, 2: PRINT "2) LASCIA"
750 INK$ = INKEY$: IF INK$ = "" THEN 750
760 IF INK$ = "1" THEN RUN
770 IF INK$ = "2" THEN CLS : END
```

so tra **0** e **1**; per tale motivo non può essere usato quasi mai direttamente.

Per ottenere un numero compreso tra **0** e **N** si ricorre alla formula...

Int (rnd * (n + 1))

...come è possibile verificare con il seguente micro-programma che visualizza 10 numeri casuali compresi tra 0 e 9.

```
10 Cls: Key Off
20 For X = 1 To 10
30 Print Int (rnd * 10)
40 Next
```

Il programma, pur funzionando correttamente, fornisce sempre la stessa sequenza di numeri, anche se si prova a spegnere e riaccendere il computer. Ciò è dovuto alla particolare "logica" della funzione che prende in esame la particolare configurazione hardware della macchina: cambiando computer, infatti, i numeri possono cambiare, pur se la sequenza rimane sempre la stessa. Provando, però, ad inserire la linea...

```
15 randomize
```

...il programma chiede un numero compreso tra - 32768 e 32767. Facendo girare più volte il programma, ci accorgiamo che il computer associa (ad **ogni** numero digitato al momento dell'input) una sequenza diversa.

Per ottenere una sequenza associata al numero 10, invece di digitare ogni volta il numero 10, possiamo modificare la linea precedente con...

```
15 randomize 10
```

```
780 BEEP: GOTO 750
790 IF A(1, 1) <> 1 THEN 280
800 IF A(2, 1) <> 2 THEN 280
810 IF A(3, 1) <> 3 THEN 280
820 IF A(4, 1) <> 4 THEN 280
830 IF A(1, 2) <> 5 THEN 280
840 IF A(2, 2) <> 6 THEN 280
850 IF A(3, 2) <> 7 THEN 280
860 IF A(4, 2) <> 8 THEN 280
870 IF A(1, 3) <> 9 THEN 280
880 IF A(2, 3) <> 10 THEN 280
890 IF A(3, 3) <> 11 THEN 280
900 IF A(4, 3) <> 12 THEN 280
910 IF A(1, 4) <> 13 THEN 280
920 IF A(2, 4) <> 14 THEN 280
```

```
930 IF A(3, 4) <> 15 THEN 280
940 LOCATE 25, 1: PRINT "PREMI UN TASTO           ";
950 IF INKEY$ = "" THEN 950
960 SCREEN 0: WIDTH 80
970 LOCATE 10, 2: PRINT "MOSSE EFFETTUATE:"; M
980 GOTO 730
```

Gioco del 15, continuazione e fine

Per esigenze di impaginazione, sulla rivista il listato sembra spezzettato in più parti. Inutile dire che, invece, è unico; alcune righe sono numerate in neretto per far capire meglio il loro inizio e la loro fine.

Per fare in modo che il computer generi, di volta in volta, una sequenza di numeri diversa dalla precedente (senza alcun intervento dell'operatore) possiamo usare la funzione **Timer** che restituisce il numero di secondi trascorsi dalla mezzanotte o dal reset di sistema. Con...

```
15 randomize timer
```

...otteniamo un programma che, ogni volta che viene attivato, stampa una sequenza di numeri casuali diversa dalla precedente.

Grafica con Get e Put

L'istruzione Get consente di memorizzare, in un vettore, un'area rettangolare di schermo che può essere riscritta con l'istruzione Put.

In questo modo è possibile ottenere animazioni ad alta velocità o registrare determinate aree di schermo.

Per prima cosa analizziamo le dimensioni del vettore. Due byte contengono quella **orizzontale**, due quella **verticale** mentre i rimanenti byte sono destinati a contenere i **dati** sull'area di schermo.

Il numero di byte necessari per memorizzare un'area di schermo di dimensioni X e Y è dato dalla formula:

```
N =4 +Int ((x * B + 7) /8) *Y
```

...in cui **B** vale **1** per la grafica in due colori e **2** per la grafica in quattro colori.

La formula per determinare il numero di elementi necessari per memorizzare N byte è:

```
E = Int ( (n + 7) / 8)
```

...se il vettore ha elementi in doppia precisione, mentre...

```
E = Int ( (n + 3) / 4)
```

...se il vettore ha elementi in singola precisione, e...

```
E = Int ( (n + 2) / 2)
```

...se il vettore ha elementi interi.

```
1 REM Esempio di Random
10 CLS : KEY OFF: DIM Z%(9)
20 SCREEN 1
30 LOCATE 1, 1: PRINT "A FUNZIONE AND"
40 GET (0, 0)-(7, 7), Z%
50 FOR X = 312 TO 0 STEP -1
60 PUT (X, 0), Z%, PSET
70 NEXT
80 FOR X = 0 TO 312
90 PUT (X, 0), Z%, PSET
100 NEXT
110 LOCATE 2, 1: PRINT "PREMI UN TASTO"
120 INK$ = INKEY$: IF INK$ = "" THEN 120
130 CLS
140 LOCATE 1, 1: PRINT "X FUNZIONE XOR"
150 GET (0, 0)-(7, 7), Z%
160 FOR X = 0 TO 312
170 PUT (X, 0), Z%, XOR
180 PUT (X, 0), Z%, XOR
190 NEXT
200 FOR X = 312 TO 0 STEP -1
210 PUT (X, 0), Z%, XOR
220 PUT (X, 0), Z%, XOR
230 NEXT
240 LOCATE 2, 1: PRINT "PREMI UN TASTO"
250 INK$ = INKEY$: IF INK$ = "" THEN 250
260 CLS
270 SCREEN 0: WIDTH 80
```

Listato che consente l'esame delle funzioni Put e Get e Rnd

Prima di dimensionare il vettore, si tenga presente che gli elementi di un vettore normalmente iniziano partendo da **0**, salvo il caso in cui non venga specificato diversamente dall'istruzione **Option Base**.

La sintassi del comando Get è:

```
Get (x1, Y1) -(x2, Y2),Array
```

...in cui **X1**, **Y1** sono le coordinate in alto a sinistra e **X2**, **Y2** le coordinate in basso a destra dell'area da memorizzare e **Array** il nome del vettore in cui sono memorizzati i dati.

La sintassi del comando Put è:

```
Put (x, Y), Array [,Action]
```

...in cui **(X, Y)** sono le coordinate in alto a sinistra dell'area in cui visualizzare Array, nome del vettore contenente l'immagine, e il parametro Action serve per effettuare operazioni logiche tra i colori dei punti da visualizzare e può assumere cinque valori:

Con **Pset** l'immagine viene disegnata sullo schermo così come è stata memorizzata con Get.

Con **Preset** svolge la stessa funzione di **Pset**, solo che l'immagine viene ridisegnata in negativo.

Con **And** sono disegnati solo i punti dell'immagine comuni allo schermo e a quella memorizzata.

Con **Or** l'immagine memorizzata viene sovrapposta a quella eventualmente esistente nell'area di schermo indicata.

Con **Xor** i punti dell'immagine memorizzata corrispondenti a punti del video vengono invertiti; riscrivendo l'immagine nello stesso punto, l'immagine scompare e il fondo resta inalterato. Questa opzione è quella di default.

Un'altra tecnica usata per effettuare animazioni è quella di lasciare intorno all'immagine un margine uguale o maggiore allo spostamento dell'immagine usando l'opzione **Pset**.

In questo modo si ottengono visualizzazioni più veloci perchè basta una sola istruzione Put; la qualità dell'animazione, inoltre, è migliore, ma bisogna memorizzare un'area più grande di quella richiesta e lo sfondo viene cancellato.

Il programma pubblicato in queste pagine chiarirà meglio le idee dal momento che è in grado di gestire le varie opzioni disponibili con Put.

A PROPOSITO DI VARIABILI

La superficialità può generare, operando in Pascal (Turbo oppure Quick), risultati inaspettati

Le prime pagine dei manuali, destinate alle informazioni di carattere generale, vengono tradizionalmente evitate con cura sia dagli esperti (che, quindi, dimostrano di non esser tali) sia dai principianti, che hanno fretta di imparare molto in poco tempo.

Ed è un vero peccato, perchè in tal modo si perde la possibilità di soffermarsi su alcuni concetti, ed esperimenti relativi, che potrebbero rivelarsi preziosi nella ricerca di successivi, (in)spiegabili errori di programmazione.

In queste brevi note ci riferiremo ad alcune prove effettuate allo scopo di vedere che cosa succede prendendo alla lettera ciò che viene riportato sui manuali originali; e, soprattutto, allo scopo di osservare ciò che accade quando si vuole esaminare in modo approfondito il comportamento del linguaggio Pascal in alcune circostanze.

Precisiamo che le prove sono state effettuate sia con il **Turbo Pascal 5.5** (*Borland*, d'ora in poi: **TP**) sia con il **QuickPascal** (*MicroSoft*, d'ora in poi: **QP**); questi, infatti, presentano caratteristiche, e risultati, praticamente identici.

Iniziamo dai manuali: a pagina 23 (TP, Guida di riferimento) e a pagina 2.2 (QP) viene riportata la tabella relativa al campo di esistenza dei **dati interi** che possono essere assegnati alle variabili di vario tipo. La tabella è pubblicata anche in questa pagina.

Sembrerebbe, ad un esame superficiale, che l'eventuale assegnazione di valori **esterni** al range (= campo di esistenza) supportato dai singoli tipi di dati, debba portare ad un errore di **Overflow** (= "straripamento" di valore). In effetti non è così, e lo dimostra l'output del primo programma (**Gestione Interi**) riportato in queste pagine.

I risultati, apparentemente bizzarri, si spiegano prendendo in esame la notazione binaria (o meglio, esadecimale) e tenendo conto che, nelle operazioni di assegnazione, il Pascal tenta di trascrivere il valore elaborato nei byte a disposizione del tipo di dato considerato.

Ad esempio, il valore decimale **-125** corrisponde a **$83** nella notazione esadecimale che tiene conto anche dei segni positivi e negativi; nella convenzione ad un solo byte (tipo **ShortInt**), però, il Pascal non tiene conto del segno ed il valore $83 viene considerato come positivo. In questo caso, quindi, a $83 viene assegnato il valore che gli compete con tali premesse (cioè **131**).

Allo stesso modo, anche se il tipo **Word** ha a disposizione due byte, il valore **-125** viene considerato pari a **65411** dal momento che la sua "traduzione" fornisce **$FF83**; questa, nella convenzione senza segno, corrisponde, appunto, a 65411. Chi ha a disposizione una calcolatrice in grado di effettuare conversioni tra valori esadecimali in decimali (e viceversa) può verificare rapidamente gli altri casi.

Si può affermare che il solo tentativo di superare il campo di esistenza del tipo **LongInt** genera un errore di overflow che, come tale, interrompe il programma in esecuzione (sinceratevene digitando, in risposta, un valore esageratamente grande).

In tutti gli altri casi, invece, il Pascal opera egualmente l'assegnazione, ma il risultato è del tutto diverso da ciò che ci si aspetta. Esperimenti analoghi possono esser compiuti operando anche sulle variabili in virgola mobile, ma la complessità del caso (dovuta soprattutto alla presenza, o meno, del processore matematico) ci costringe a rinviare la trattazione.

Si potrebbe obiettare che l'attento programmatore dovrebbe evitare la "fusione" di variabili appartenenti a tipi di dati diversi tra loro; in certi casi, tuttavia, si è presi dalla tentatazione di effettuare elaborazioni precise (ricorrendo a tipi di dati utilizzanti molti byte, o addirittura in virgola mobile) per poi "scaricarli" nelle variabili che devono obbligatoriamente essere di tipo intero (come i parametri delle istruzioni grafiche).

N. bytes	Tipo dati	Minimo	Massimo
1	Shortint	- 128	+ 127
1	Byte	0	+ 255
2	Integer	- 32768	+ 32767
2	Word	0	+ 65535
4	Longint	- 21474836	+ 2147483647

Tipi di dati interi in Pascal e realtivo range di valori

Come è veloce l'intero

Le velocità di elaborazione cambiano notevolmente a seconda del numero di bytes richiesti dai tipi di dati stessi; in pratica, possiamo affermare che (almeno nel caso, ad esempio, di nidificazioni di cicli **For...Do**) i tempi di elaborazione delle variabili tipo Byte e Shortint (che richiedono un solo byte) sono l'uno minore

```
Program Velocita;
uses dos,crt;
var
 Intero_corto, Cor1, Cor2, Cor3 :shortint;
 Bit_8, Bit1, Bit2 :byte;
 Intero, Int1, Int2 :Integer;
 Doppio_intero, Int_do1,Int_do2 :Word;
 Intero_lungo, Int_lu1, Int_lu2 :LongInt;
 tyme1, tyme2, tyme3, tyme4: word;

 begin
 clrscr; writeln ('Calcolo i tempi dei cicli:');
settime (0, 0, 0, 0);
for Intero_corto :=0 to 127 do
    begin
      for Cor1 :=0 to 127 do
        begin
          For Cor2 :=0 to 127 do
        end;
    end;
    gettime(tyme1, tyme2, tyme3, tyme4);
 writeln('ShortInt', tyme2,' ',tyme3,' ', tyme4);

settime (0, 0, 0, 0);
for Bit_8 :=0 to 127 do
  begin
    for Bit1 :=0 to 127 do
    begin
      For Bit2 :=0 to 127 do
    end;
  end;
  gettime (tyme1, tyme2, tyme3, tyme4);
 writeln('T. Byte',tyme2,' ',tyme3,' ', tyme4);

settime (0, 0, 0, 0);
for Intero:=0 to 127 do
  begin
    for Int1:=0 to 127 do
    begin
      For Int2:=0 to 127 do
    end;
  end;
  gettime (tyme1, tyme2, tyme3, tyme4);
  writeln('Integer ',tyme2,' ',tyme3,' ', tyme4);

settime (0, 0, 0, 0);
for Doppio_intero:=0 to 127 do
  begin
    for Int_do1:=0 to 127 do
    begin
      For Int_do2:=0 to 127 do
    end;
  end;
  gettime (tyme1, tyme2, tyme3, tyme4);
  writeln ('Tempo Word    ',tyme2,' ',tyme3,' ',
tyme4);

settime (0, 0, 0, 0);
for Intero_Lungo:=0 to 127 do
  begin
    for Int_lu1:=0 to 127 do
    begin
      For Int_lu2:=0 to 127 do
    end;
  end;
  gettime (tyme1, tyme2, tyme3, tyme4);
  writeln ('Longint',tyme2,' ',tyme3,' ', tyme4);

  writeln;writeln('Premi un tasto');
  repeat until keypressed;
  end.
```

dell'altro; un incremento, anche se modesto, si ha trattando variabili che richiedono due bytes; un tempo circa doppio, invece, richiedono le variabili di tipo Longint, memorizzate in quattro bytes. Il programma "Calcolo tempi", qui pubblicato, determina il tempo impiegato da un calcolatore AT-compatibile (10 Mhz) per elaborare l'intero programma.

Altre note degne di rilievo, nel pur semplice programma "Gestione Interi", sono relative all'uso dell'unità **CRT**, indispensabile per l'utilizzo della procedura **ClrScr** (che cancella lo schermo) e **ReadKey** (che attende la pressione di un tasto per associarne il codice ad una variabile di tipo String oppure Char). Da sottolineare l'uso di adeguati caratteri, posti in coda al nome delle variabili, per ricordare, in ogni momento, il tipo di dato con cui si ha a che fare. Nel linguaggio

```
125 Appartiene al range SHORTINT
125 Appartiene al range BYTE
125 Appartiene al range INTEGER
125 Appartiene al range WORD

Tu hai scritto -100000
Ma io lo leggo come SHORTINT 96 !
Ma io lo leggo come BYTE      96 !
Ma io lo leggo come INTEGER   31072 !
Ma io lo leggo come WORD     31072 !

-125 Appartiene al range SHORTINT
Ma io lo leggo come BYTE       131 !
-125 Appartiene al range INTEGER
Ma io lo leggo come WORD      65411 !

Tu hai scritto 40000
Ma io lo leggo come SHORTINT 64 !
Ma io lo leggo come BYTE       64 !
Ma io lo leggo come INTEGER   -25536 !
40000 Appartiene al range WORD
```

Output del programma "Gestione Interi" in risposta ai valori 125, -125, 40000, -100000.

```
PROGRAM Esperimenti_su_interi (Tipi di dati interi )};
  {Compatibile TURBO PASCAL (Borland); QUICKPASCAL (Microsoft)}
uses crt; {Necessario per CLRSCR}
Var
Intero_Corto        :Shortint; {      -128  + 127}
Bit_8               :Byte;     {         0  + 255}
Intero              :Integer;  {    -32768  + 32767}
Doppio_intero       :Word;     {         0  + 65535}
Intero_Lungo        :Longint;  {-21474836   + 2147483647}
Dato_da_esaminare   :Longint;
Tasto    :char;
Begin {1}
Tasto:=chr(13); clrscr;
while Tasto = chr(13) do
begin {2}
 writeln;
 writeln ('1-  :Shortint  {intero tra    -128           +127}');
 writeln ('2-  :Byte;     {  "    "         0           +255}');
 writeln ('3-  :Integer;  {  "    "    -32768        + 32767}');
 writeln ('4-  :Word;     {  "    "         0        + 65535}');
 writeln ('5-  :Longint;  {  "    " -21474836   + 2147483647}');
 writeln;
 write ('Digita un val. comp. tra -21474836 e + 2147483647 ');
 readln (Dato_da_esaminare); writeln;
 Intero_corto := Dato_da_esaminare;
 if (Dato_da_esaminare  <-128) or (Dato_da_esaminare  >127) then
      begin {3}
         writeln ('Tu hai scritto ',Dato_da_esaminare);
         writeln ('Io leggo come SHORTINT ', Intero_corto,' !');
      end {3}
         else writeln(Intero_Corto,' E' del range SHORTINT');
     Bit_8 := Dato_da_esaminare;
    if (Dato_da_esaminare  <0) or (Dato_da_esaminare  >255) then
      begin {4}
         writeln('Ma io lo leggo come BYTE      ', Bit_8,' !');
      end {4}
         else writeln(Bit_8,' Appartiene al range BYTE');
     Intero := Dato_da_esaminare;
 if (Dato_da_esaminare <-32768)or(Dato_da_esaminare >32767) then
      begin {5}
         writeln('Ma io lo leggo come INTEGER   ',Intero ,' !');
      end {5}
         else writeln(Intero,' Appartiene al range INTEGER');
     Doppio_Intero := Dato_da_esaminare;
  if (Dato_da_esaminare  <0) or (Dato_da_esaminare  >65535) then
      begin {6}
         writeln ('Io leggo WORD     ',Doppio_Intero ,' !');
      end {6}
        else writeln(Doppio_Intero,' Appartiene al range WORD');
writeln;
writeln ('Return per continuare, altro tasto per finire');
tasto:=readkey;clrscr;
end; {2}
end. {1}
```

Il programma Gestione Interi

Calcolo i tempi dei cicli

```
                    Sec. Cent.
Tempo ShortInt:   8    56
Tempo Byte    :   8    18
Tempo Integer :   8    84
Tempo Word    :   8    18
Tempo Longint :  15    81
```

Pascal, infatti, non sono previsti (come nel Basic) particolari suffissi per definire una variabile; ne consegue che un qualsiasi nome può indicare un qualsiasi tipo di variabile. L'apparente libertà d'azione, tuttavia, pregiudica la leggibilità di un programma, per giunta poco commentato, se questo viene riesaminato dopo molto tempo per eventuali modifiche. Imponendo invece, fin dall'inizio, un "suffisso" in grado di ricordare al programmatore il tipo di dato usato (esempio: Nome_**st**, Peso_**re**, Conta_**by**, Tasto_**ch**, rispettivamente per variabili di tipo **S**tringa, **Rea**le, **By**te e **Ch**ar) le probabilità di cadere in errore diminuiscono.

Si noti, inoltre, l'utilizzo delle parentesi graffe ({ }) in cui sono racchiusi la successione dei numeri dei cicli **Begin / end** aperti e chiusi. Di solito, infatti, l'*indentazione* dovrebbe esser sufficiente per individuare facilmente, nel listato, i "confini" di un ciclo Begin / end.

Tuttavia, in molti casi, l'individuazione dell'inizio e della corrispondente fine possono dare adito a dubbi. L'utilizzo insolito del commento, cioè **begin {1}** seguito dal corrispondente **end {1}** rende più semplice la leggibilità del listato stesso, soprattutto nel caso di "nidificazioni"; a meno, ovviamente, di errori commessi dal programmatore durante l'annotazione delle stesse parentesi graffe...

Importante, soprattutto per i principianti, sottolineare la necessità di rispettare la sintassi **If... Then... Else** che, in caso di errata trascrizione, non sempre genera un errore in fase di compilazione; nei casi più semplici, come quelli pubblicati, si rinuncia volentieri all'uso di Begin / End per indicare l'elaborazione da seguire nel caso l'espressione risulti vera. Tuttavia in seguito, per migliorare l'aspetto estetico del programma, spesso si "spezza" la riga contenente **If**, con il risultato che l'elaborazione prosegue in modo indesiderato. Utilizzando Begin / end, invece, tale possibilità non si verifica, nemmeno

disponendo i vari comandi su più righe video.

Da notare, infine, l'utilizzo di **While**, accoppiato alla variabile di tipo carattere **char**, che consente di "uscire" dal programma solo nel caso in cui, terminata l'elaborazione, si prema un tasto qualsiasi diverso da Return.

Per ciò che riguarda, invece, il programma incaricato di evidenziare la velocità di elaborazione, diremo che l'unità **Dos** serve per la gestione delle variabili legate al trascorrere del tempo (Settime, Gettime), che sono di tipo Word.

Attenti allo Stack

Una delle caratteristiche più interessanti dei linguaggi ad alto livello (tra cui il Pascal) è certamente la **ricorsività**, vale a dire la possibilità, da parte di una procedura o funzione, di richiamare se stessa.

Il programma **Count Down** evidenzia tale potenzialità, ed è di valore esclusivamente didattico dal momento che si potrebbe realizzare un conteggio alla rovescia senza scomodare la ricorsività.

Le variabili usate sono tutte di tipo Real e la *funzione* **Conta_re**, anch'essa di tipo Real, utilizza la variabile **Contatore_re** che viene decrementata finchè risulta maggiore di zero. Lo stesso risultato si può ottenere con un ciclo While, ma lasciamo al lettore, per esercizio, le modifiche del caso.

Lanciato il programma, e digitando un valore in risposta alla domanda relativa al conteggio desiderato, sullo schermo apparirà il count down, formattato secondo il comando presente in Write (cioè 4 cifre significative, nessuna cifra decimale).

Sembrerebbe un programma di nessuna utilità didattica, a parte quella di esaminare da vicino una ricorsione; al contrario, il listato si presta per verificare un fenomeno poco noto ai principianti in materia di ricorsione. Rispondendo con un valore sufficientemente elevato (con il nostro elaboratore, questo era **991**) il conteggio inizia regolarmente, ma si blocca, poco dopo, con l'emissione del messaggio **"Stack Overflow Error"**. Ciò dimostra che... nulla si crea e nulla si distrugge; la ricorsione può sussistere a patto che i successivi "richiami" della funzione vengano memorizzati da qualche parte (cioè nello Stack). Eseguito l'ultimo richiamo, infatti, lo stack viene via via "scaricato".

Per rendersi conto dell'effettivo funzionamento del programma, si abiliti, con il tasto **F6**, l'opzione **Switch** (ci riferiamo a **TP**) e, dopo aver premuto Return, si digiti il nome della variabile che vogliamo controllare (cioè: **Contatore_co**). A questo punto, dopo aver premuto ancora F6 per "uscire" dalla finestra **Watch**, si faccia partire il programma con il tasto **F7** (funzione **Trace**) in modo da esaminare, passo dopo passo, ciò che accade.

Al momento della domanda relativa al conteggio, rispondete con un valore basso (esempio: **8**) e premete Return. Premendo successivamente F7, noterete alcuni interessanti avvenimenti: anzitutto, ad ogni pressione di F7 il programma elabora **Begin {1 fun.}**, controlla l'**If** posto subito dopo e conclude con **End {2 fun.}** senza mai "chiudere" il ciclo **End {1 fun.}**. E' in questa fase che lo stack viene riempito, passo dopo passo. Contemporaneamente si può notare come il valore attribuito alla variabile Contatore_re (visualizzata nella finestra di Watch) viene, giustamente, decrementato. Il lettore troverà ulteriore conferma con **Alt F5**, in modo da esaminare ciò che viene riportato sullo schermo.

Ma attenzione: non appena il conteggio arriva al valore nullo (Contatore_re = 0) sarà necessario premere 8 volte il tasto F7 per "scaricare" lo Stack; è in questo momento, infatti, che la ricorsione cessa di essere attiva e Pascal recupera i dati dello stack. Si potrà notare, contemporaneamente, che il valore attribuito a Contatore_re **aumenta** ad ogni ciclo.

Completato lo svuotamento dello stack il programma termina, non prima di aver attribuito, alla variabile Contatore_re, un valore... inaspettato.

Eliminando le parentesi graffe all'istruzione Write (posta prima dell a fine della funzione **end {1 fun}**), il lettore potrà esaminare ciò che succede al "ritorno" dalla ricorsione.

E' utile concludere, quindi, che una ricorsione non può agire all'infinito, ma bisogna fare i conti con l'area disponibile nello Stack; inoltre sarebbe opportuno effettuare controlli sulle variabili utilizzate in cicli ricorsivi, dopo che questi hanno termine.

Nel listato, l'unica novità di rilievo è rappresentata dall'aver distinto i cicli Begin / End, appartenenti alla funzione, con il termine **Fun {1 fun.}** in modo da distinguerli dai cicli del programma principale.

```
Program Conto_alla_rovescia;
{Usa Trace per osservare l'effetto!}
uses crt;
var
 Contatore_re, cont_re :real;
 tasto_ch              :char;

function Conta_re (Contatore_re :real) :real;
  begin {1 fun.}
   if (Contatore_re > 0) then
     begin {2 fun.}
      write (Contatore_re:4:0,' ');
       cont_re := Conta_re (Contatore_re - 1)
     end  {2 fun.}
     else writeln ('Fine');
    {write (contatore_re: 4: 0, ' ');}
    end; {1 fun.}

begin {1}
   clrscr; Tasto_ch :=chr(13);
while Tasto_ch = chr(13) do
  begin {2}
write ('Numero < 1000 '); readln (Contatore_re);
Contatore_re := Conta_re (Contatore_re);
writeln; writeln ('Return = cont; tasto = fine');
         tasto_ch :=readkey;
   end; {2}
end. {1}
```

by Valerio & Fabio Capello

Sfida musicale, ecco una risposta

La sfida di tipo "musicale" lanciata sul N. 81 non è rimasta inascoltata...

Fabio! Hai letto l'ultimo numero di C.C.C., quello che ho appena comprato ?". *"No, perché?"*. "Guarda a pagina 86, c'è una sfida ai lettori nella quale si chiede la creazione di un programma che gestisca la tastiera del computer più o meno come quella di un pianoforte...". *"Ma un programma del genere c'è già sul manuale del C/64!"*. "Lasciami finire! Si chiede inoltre che il programma controlli se la nota suonata è in disaccordo con le altre e, nel caso, la corregga."...

Con questa discussione iniziò una lotta di vari giorni con la quale ho cercato di staccare mio fratello da Kick Off e dirottarlo su un W.P. per scrivere un qualcosa che mi consentisse di creare il programma richiesto da Commodore Computer Club. Siamo infatti due fratelli che si interessano l'uno (Fabio) di musica e l'altro (io, Valerio) di programmazione (oltre che di skateboard).

Alla fine fu stesa una "relazione" nella quale viene spiegato come una nota può risultare "stonata" rispetto alla successiva (o precedente).

Fu il punto di partenza per la realizzazione del programma richiesto, sviluppato in **Basic v 2.0 per C/64**, pubblicato in queste pagine.

Alla fine di queste note, ovviamente, verrà descritto l'algoritmo usato anche se, per quanto riguarda la parte generale del programma (che si occupa del calcolo delle frequenze e del riconoscimento delle note associate ai tasti) questa ricalca quella del listato dimostrativo che compare a pagina 147 del manuale del C/64 (appendice J).

Tonalità e scale maggiori

La presenza di note in "disaccordo" con altre presenti all'interno di una composizione musicale può essere spiegata considerando la **tonalità** nella quale il brano stesso viene composto.

Esistono, infatti, diverse tonalità che differiscono da quella (in genere più conosciuta) che segue la scala naturale di **DO maggiore** e che corrisponde alla serie di *sette note* ottenuta con la pressione, in sequenza, dei *tasti bianchi* del pianoforte partendo da una qualsiasi nota DO.

Tra una nota e l'altra esistono, infatti, intervalli ben precisi (detti **toni** o **semi-toni** a seconda che la distanza sia **maggiore** o **minore** rispetto al "grado" precedente) e che, se non rispettati, *possono* creare una dissonanza.

Le "distanze" si possono individuare nella scala di DO maggiore come appare nello schema pubblicato, dove con **1** si indica l'intervallo di un tono, con **1/2** l'intervallo di un semi-tono.

Per rispettare tali distanze anche nel momento in cui si compone in una tonalità diversa da quella in DO, bisogna far ricorso alle alterazioni: il **diesis** (**#**), che *solleva* di un semi-tono il grado a cui si lega; il **bemolle** (**b**) che *abbassa* di un semi-tono il grado a cui si lega; il **bequadro** che, nel caso in cui ci si trova in una scala diversa da quella in DO, riporta le note che risulterebbero altrimenti alterate al loro stato naturale; infine, raramente, il **doppio** diesis ed il doppio bemolle che abbassano o sollevano di un tono la nota alla quale vengono associati.

Se vogliamo comporre in una tonalità differente da quella in DO dobbiamo quindi inserire, tra le note che formano la nuova scala, alcune alterazioni che permettano di rispettare le distanze indicate nello schema precedente.

Se si intende comporre in tonalità di SOL, ad esempio, bisognerà procedere nel seguente modo: si scrivono gli **otto** gradi che compongono la scala di SOL partendo, appunto, dalla nota SOL; si confrontano gli intervalli che intercorrono tra essi basandosi sullo schema precedente.

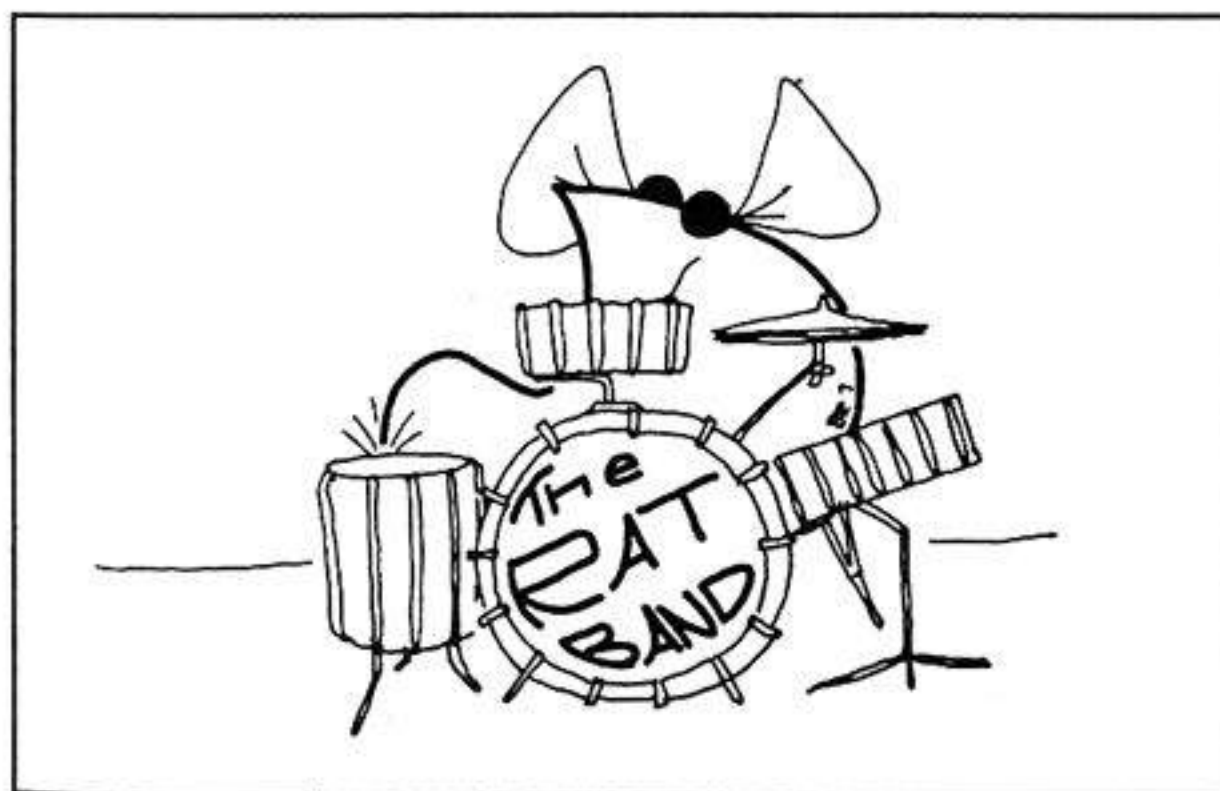
Si agisce, quindi, alterando le note tra le quali non vi è la distanza che rispetta

1		2		3		4		5		6		7		8
do		re		mi		fa		sol		la		si		do

Tabella 1: la scala del Do

1		2		3		4		5		6		7		8
sol		la		si		do		re		mi		fa		sol
	1		1		1/2		1		1		1/2		1	

Tabella 2: la scala del Sol

1		2		3		4		5		6		7		8
sol		la		si		do		re		mi		fa#		sol
	1		1		1/2		1		1		1		1/2	

Tabella 3: scala in Sol maggiore

quella della scala in DO maggiore. In questo caso, tra il *sesto* ed il *settimo* grado, vi è un semi-tono anziché uno intiero (vedi scala del DO), e tra il settimo e lo ottavo c'è un tono intero invece che mezzo.

Possiamo quindi modificare il settimo grado **FA**, *sollevandolo* di un semi-tono ed ottenendo un **FA#** in modo da portare ad un tono intero la distanza tra MI e FA (infatti 1/2 tono più un altro 1/2 tono, ottenuto sollevando FA, è uguale ad 1 tono intiero); oppure *abbassando* di un semi-tono l'intervallo tra FA e SOL (dal momento che se ad un tono togliamo 1/2 tono, ottenuto "allargando" lo spazio tra i due gradi precedenti, si ha 1/2 tono).

La scala di SOL ottenuta con l'aggiunta del FA# è, ora, del tutto uguale (pur se limitandoci a considerare gli intervalli tra nota e nota) a quella naturale di DO, come si può notare dalla terza tabella.

Si ottiene, quindi, la cosiddetta scala di SOL maggiore.

Per semplificare il passaggio dalla scala naturale di DO maggiore alle altre scale maggiori, riportiamo la tabella 4 che indica le note da alterare nelle varie tonalità nelle quali si intende lavorare, tenendo presente che nell'ultima scala è riportato il MI# che corrisponde al FA, per evitare la presenza di quest'ultimo assieme al FA# stesso.

Quando si deve comporre un pezzo in una certa tonalità, bisognerà sostituire, alla nota naturale, quella\e dello schema che corrisponde\ono a tale tonalità.

La nota giusta

La tabella è facilmente ricavabile da un semplice ragionamento. Bisogna, anzitutto, tener presente che la *tonica* è il **primo grado** della scala: nel caso della scala di DO, per esempio, è il Do; per la tonalità di SOL, invece, è il Sol.

Bisogna partire, per quanto riguarda le scale, dalla tonica della scala di DO e contare **cinque** note arrivando, quindi, alla tonica della scala di SOL, cioè:

1) Do, **2**) Re, **3**) Mi, **4**) Fa, **5**) Sol

Dal SOL al RE, analogamente...

1) Sol, **2**) La, **3**) Si, **4**) Do, **5**) Re

...e così via, sostituendo, però, al FA il FA#.

Per quanto concerne le note da alterare legando loro un **diesis**, basterà ricordare che la scala di DO maggiore non ha alterazioni, in quanto scala naturale dalla quale derivano le altre; mentre a partire dalla scala di SOL si dovrà alterare una nota partendo dal FA# e via via aggiungendo un'altra nota da alterare contando cinque gradi da quella precedente.

Si avrà così, dal FA#, il DO#:

1) Fa#, **2**) Sol#, **3**) La#, **4**) Si, **5**) Do#

Dal DO#, il SOL#:

1) Do#, 2) Re#, 3) Mi#, 4) Fa#,5) Sol

...e così via.

Ricordate che la tabella riportata è valida solo per le scale maggiori che seguono, come già detto, la scala naturale di DO. Esistono infatti anche scale minori che, invece, seguono quella naturale di **LA minore.**

Le note naturali (Do, Re, Mi, Fa, Sol, La, Si) compaiono solo in tali scale naturali (scala di DO e di LA minore, appunto); nelle altre ci sarà almeno una nota alterata. Pertanto, quando si parla di riportare una certa nota allo stato naturale, legando ad essa il bequadro, nell'eseguirla non si dovrà tener conto della scala cui essa appartiene.

Infine prestate attenzione al fatto che, in una scala, il numero complessivo di semi-toni, da una tonica all'altra, è **12** (cioè **6** toni). Se il numero dei semi-toni dovesse risultare maggiore di 12 vi è sicuramente un errore.

Riassumendo

Il paragrafo precedente spiega, dunque, quando una nota si può definire "stonata" rispetto ad un'altra.

In pratica, scegliendo di comporre una musica in una certa scala, si possono usare solo certe note, relative alla scala stessa. Le note utilizzabili nelle scale maggiori sono riportate nell'apposita tabella.

Il programma

Il programma EasyComposer chiede, all'inizio, quale scala si vuole usare, quindi si limita a controllare se la nota relativa al tasto premuto faccia parte della scala precedentemente scelta. In caso contrario la **innalza** di un semitono, se era normale (trasformandola in **diesis**) o **abbassandola**, sempre di un semitono, se era in **diesis**.

Il programma mostra un output suddiviso in **due colonne**: in quella di sinistra viene visualizzata la nota scelta, mentre in quella di destra la nota effettivamente suonata, evidenziata da una freccia se è stata "corretta".

EasyComposer utilizza due ottave per comporre la musica, associate ai tasti in modo da simulare, per quanto possibile, la posizione dei tasti bianchi e neri su un pianoforte. Per la prima ottava avremo...

```
 2 3   4 5 6
Q W E R T Y U
```

...e per la seconda:

DO	:	
SOL	:	FA#
RE	:	FA# DO#
LA	:	FA# DO# SOL#
MI	:	FA# DO# SOL# RE#
SI	:	FA# DO# SOL# RE# LA#
FA#	:	FA# DO# SOL# RE# LA# MI#

Tabella 4

DO	:	DO	RE	MI	FA	SOL	LA	SI
SOL	:	DO	RE	MI	FA#	SOL	LA	SI
RE	:	DO#	RE	MI	FA#	SOL	LA	SI
LA	:	DO#	RE	MI	FA#	SOL#	LA	SI
MI	:	DO#	RE#	MI	FA#	SOL#	LA	SI
SI	:	DO#	RE#	MI	FA#	SOL#	LA#	SI
FA#	:	DO#	RE#	MI#	FA#	SOL#	LA#	SI

Note utilizzabili nelle scale maggiori

```
 S D   G H J
Z X C V B N M
```

Un altro tasto riconosciuto da EasyComposer è la *barra spaziatrice* che, se premuta, consente di scegliere un'altra scala al posto di quella selezionata inizialmente.

Se qualcuno preferisse usare i tasti consigliati nella sfida del N. 81 di CCC, non deve fare altro che modificare la stringa **K$**, lasciando, all'inizio e alla fine della stringa stessa, due caratteri "jolly", ed eventualmente modificare le righe **Data** contenenti i nomi delle note.

I caratteri **jolly** (due punti esclamativi) servono per escludere dalla lista dei tasti musicali la prima e l'ultima nota nella lista dei Data, che servono solo per l'eventualità che la seconda e la penultima nota vengano abbassate o alzate di un semitono.

Miglioramenti

Se EasyComposer vi sembra troppo lento, potete compilarlo con **Austro-Compiler**. Il risultato finale è (quasi) perfetto, tranne per il fatto che l'istruzione **Tab** non viene accettata dal compilatore e le due colonne risulteranno leggermente disordinate. Inoltre potete fare in modo che, come in un vero pianoforte, la durata di emissione del suono sia pari alla durata della pressione del tasto.

Per ottenere tale effetto, provate a procedere in questo modo: abilitate la ripetizione automatica dei tasti (sul C/64 si ottiene con **Poke 650, 128** e **Poke 198, 1** che riduce a un solo carattere il buffer di tastiera), poi inserite il ritardo della ripetizione dei tasti al minimo (con **Poke 651, 1**) e l'incremento della ripetizione al massimo (con **Poke 652, 255** ma la locazione 652 è read-only, cioè non modificabile); infine, trovate un valore di ritardo (variabile **D**) adeguato, effettuando vari tentativi.

```
100 rem easycomposer v1.1 (c64-basic v2 version)
110 :
120 poke53280,0:poke53281,0
130 print"♥SReasycomposer v1.1R"
140 print"3(c) febbraio 1991 3free boys soft"
150 print"written by valerio capello"
160 print"music consultant: fabio capello"
170 :
180 print"QQQattendi un attimo..."
190 s=54272:rem indirizzo del sid (sound interface device)
200 fori=0to28:pokes+i,0:next:rem azzera i registri del sid
210 :
220 k$="!q2w3er5t6y7uzsxdcvgbhnjm!":rem tasti musicali
230 kn=len(k$):rem numero tasti musicali
240 dim f(kn),k(255),nn$(kn),na$(7,7)
250 :
260 rem legge le note suonabili da tastiera
270 :
280 restore:fori=1tokn:readnn$(i):next
290 data si   -1,do  ,do #,re  ,re #,mi  ,fa  ,fa #,sol ,sol#,la  ,la #,"si  "
300 data do   +1,do # +1,re   +1,re # +1,mi   +1,fa   +1,fa # +1
310 data sol  +1,sol# +1,la   +1,la # +1
320 data si   +1,do   +2
330 :
340 rem legge le tabelle delle scale maggiori
350 :
360 fori=1to7:forj=1to7:readna$(i,j):next:next
370 data do  ,re  ,mi  ,fa  ,sol ,la  ,"si  "
```

```
380 data do  ,re  ,mi  ,fa #,sol ,la  ,"si  "
390 data do #,re  ,mi  ,fa #,sol ,la  ,"si  "
400 data do #,re  ,mi  ,fa #,sol#,la  ,"si  "
410 data do #,re #,mi  ,fa #,sol#,la  ,"si  "
420 data do #,re #,mi  ,fa #,sol#,la #,"si  "
430 data do #,re #,mi #,fa #,sol#,la #,"si  ":rem mi# = fa
440 :
450 rem calcola le frequenze e le memorizza in f()
460 :
470 f1=7939:fori=1tokn:f(kn+1-i)=f1*5.8+30:f1=f1/2↑(1/12):next
480 fori=1tokn:k(asc(mid$(k$,i)))=i:next
490 m=1:d=200:rem m=ottava,d=durata
500 :
510 rem chiede la tonalita' e mostra il significato dei tasti
520 :
530 pokes+5,0:pokes+6,230:pokes+24,15:rem adsr per voce#1 e volume on
540 print"scegli la scala:"
550 print"1-do 2-sol 3-re 4-la 5-mi 6-si 7-fa#":inputt$:t=val(t$)
560 ift<1ort>7then540
570 print"premete questi tasti per suonare:"
580 fori=2tokn-1:poke212,1:printmid$(k$,i,1);:next:poke212,0:print
590 print"o spazio per cambiare scala"
600 :
610 rem attende che venga premuto un tasto fra quelli ammessi
620 :
630 geta$:ifa$=""then630
640 ifa$=" "then540
650 k=0:fori=2tokn-1:ifa$=mid$(k$,i,1)thenk=1
660 next
670 ifk=0then630
680 :
690 z=k(asc(a$)):gosub770
700 fr=f(z)/m:rem calcola la nota da suonare
710 pokes,fr-256*int(fr/256):pokes+1,fr/256:pokes+4,17:rem suona la nota giusta
720 fori=1tod:next:pokes+4,16:rem ciclo di ritardo per la durata della nota
730 goto630
740 :
750 rem controlla se la nota scelta e' giusta
760 :
770 ns$=nn$(z):printns$;tab(12);:nc$=left$(ns$,4)
780 :
790 rem controlla se la nota scelta e' presente nella tabella
800 rem della scala attualmente selezionata
810 ok=0:fori=1to7:ifna$(t,i)=nc$thenok=1
820 next
830 :
840 rem se si, torna al programma principale
850 ifok=1thenprintns$:goto940
860 :
870 rem se no, prima di tornare al programma principale
880 rem corregge la nota scelta, abbassandola di un
890 rem semitono se e' un diesis(#) o alzandola
900 rem di un semitono in caso contrario
910 ifright$(nc$,1)="#"thenz=z-1:goto930
920 z=z+1
930 printnn$(z);tab(20)"<---"
940 return

ready.
```

di Domenico Mobilia

Amiga risponde alla sfida musicale

Un programma scritto in Amos è una valida occasione per valutare le caratteristiche di altri interpreti Basic, specifci per Amiga.

Il lettore Domenico Mobilia (di **Anoia S.re, RC**) ha voluto contribuire con una sua "creazione" alla sfida del mese relativa al generatore musicale (di cui è presente, su questo stesso numero, la versione per **C/64**). Purtroppo non ha voluto seguire il consiglio di telefonarci **prima** di inviare il dischetto, altrimenti avremmo pubblicato per intero il suo interessante articolo (**se** lo avesse accluso sul floppy, e **non** solo su carta..). Il programma è scritto in **Amos**, il potente interprete per Amiga (a proposito, abbiamo deciso di pubblicare alcuni articoli sul suo corretto utilizzo; non vi sembra un'idea interessante?).

Una spiegazione sul funzionamento del programma, pur se sommaria, è d'obbligo.

```
'GENERATORE MUSICALE BY MIMMO MOBILIA 1991
'  DATI VIDEO
Curs Off : Hide : Cls 0 : Screen Open 1,660,300,16,Hires
: Curs Off : Hide : Cls 0 : Paper 0
Locate 20,2 : Pen 13 : Print Border$(" G E N E R A T O R
E   M U S I C A L E ",2)
Locate 10,12 : Pen 12 : Print Border$("NOTA
CORRISPONDENTE",1)
Locate 10,17 : Pen 12 : Print Border$("OTTAVA",1)
Wait Vbl
'DIMENSIONAMENTO E AZZERAMENTO VARIABILI
Dim N(249),N$(249),X(500)
X(1)=0 : I=0 : K=0 : O=1
'CARICAMENTO VETTORI
N(97)=1 : N(119)=2 : N(115)=3 : N(101)=4 : N(100)=5 :
N(102)=6 : N(116)=7 : N(103)=8 : N(121)=9 : N(104)=10 :
N(117)=11 : N(106)=12 : N(107)=13 : N(111)=14 :
N(108)=15 : N(112)=16
N$(0)="SI" : N$(1)="DO" : N$(2)="DO#" : N$(3)="RE" :
N$(4)="RE#" : N$(5)="MI" : N$(6)="FA" : N$(7)="FA#" :
N$(8)="SOL" : N$(9)="SOL#"
N$(10)="LA" : N$(11)="LA#" : N$(12)="SI" : N$(13)="DO" :
N$(14)="DO#" : N$(15)="RE" : N$(16)="RE#" : N$(17)="MI"
N$(97)="DO" : N$(119)="DO#" : N$(115)="RE" :
N$(101)="RE#" : N$(100)="MI" : N$(102)="FA" :
N$(116)="FA#" : N$(103)="SOL" : N$(121)="SOL#" :
N$(104)="LA" : N$(117)="LA#" : N$(106)="SI" :
N$(107)="DO" : N$(111)="DO#" : N$(108)="RE" :
N$(112)="RE#"
'ROUTINE DI INPUT/1
Locate 17,17 : Pen 15 : Print O : Rem stampa ottava
sullo schermo
Do
   10 A$=Inkey$ : If A$="" Then 10
```

Come gira

Agli elementi del vettore **N** sono associati i tasti premuti; ovviamente il vettore è sovradimensionato dal momento che le file di tasti consentiti sono solo due (e nemmeno complete). Tale tecnica, tuttavia, consente di individuare facilmente il tasto premuto ed agire di conseguenza. Ad esempio, ad **N(97)** corrisponde il tasto **"A"** (97 è infatti il codice Ascii di "A") e così per ogni tasto "lecito".

Il vettore **N$**, in modo analogo, memorizza le note stampabili a video.

Con la routine **Input/1** è possibile attivare l'ottava (selezionabile in qualsiasi momento agendo sui tasti di spostamento del cursore).

Con la routine **Input/2** viene attivato il processo logico di eliminazione della nota stonata; se, infatti, l'ultima nota digitata risulta non facente parte della scala maggiore calcolata dal programma (elaborata in base alla prima nota suonata), essa sarà sostituita con quella più "vicina" (sempre facente parte della scala maggiore).

Per esser più precisi, la nota emessa sarà più alta di quanto digitato per errore

```
    If A$=Chr$(30) and K<72 Then K=K+12 : O=O+1 : Locate
17,17 : Pen 15 : Print O : Goto 10 : Rem incremento e
stampa ottave
    If A$=Chr$(31) and K>11 Then K=K-12 : O=O-1 : Locate
17,17 : Pen 15 : Print O : Goto 10 : Rem decrementa e
stampa ottave
    Locate 18,25 : Pen 0 : Print Space$(48) : Rem questa
riga cancella dallo schermo la 14
    If A$=Chr$(30) Then Goto 10 : Rem questa riga elimina
il messaggio d'errore causato dalla pressione del tasto
freccia/alto
    If A$=Chr$(31) Then Goto 10 : Rem questa riga elimina
il messaggio d'errore causato dalla pressione del tasto
freccia/basso
    Locate 31,12 : Pen 0 : Print Space$(26) : Rem questa
riga cancella dallo schermo la successiva
    If N(Asc(A$))<1 or N(Asc(A$))>16 Then Locate 31,12 :
Pen 0 : Print Space$(26) : Locate 31,12 : Pen 3 : Print
"TASTO INABILITATO AL SUONO" : Goto 10
    Locate 31,12 : Print Space$(4) : Pen 6 : Locate 31,12
: Print N$(Asc(A$)) : Rem stampa note sullo schermo
    'ROUTINE DI INPUT/2
    11 I=I+1
    If I=1 Then X(1)=N(Asc(A$)) : Goto 50 : Rem memorizza
la prima nota
    12 X(I)=N(Asc(A$))
    13 If X(I)=(X(1)) Then Goto 50
    If X(I)=(X(1)+2) Then Goto 50
    If X(I)=(X(1)+4) Then Goto 50
    If X(I)=(X(1)+5) Then Goto 50
    If X(I)=(X(1)+7) Then Goto 50
    If X(I)=(X(1)+9) Then Goto 50
    If X(I)=(X(1)+11) Then Goto 50
    If X(I)=(X(1)+12) Then Goto 50
    If X(I)=(X(1)+14) Then Goto 50
    If X(I)=(X(1)+16) Then Goto 50
    If X(I)=(X(1)-1) Then Goto 50
    If X(I)=(X(1)-3) Then Goto 50
    If X(I)=(X(1)-5) Then Goto 50
    If X(I)=(X(1)-7) Then Goto 50
    If X(I)=(X(1)-8) Then Goto 50
    If X(I)=(X(1)-10) Then Goto 50
    If X(I)=(X(1)-12) Then Goto 50
    If X(I)=(X(1)-13) Then Goto 50
    If X(I)=(X(1)-15) Then Goto 50
    If X(I)=(X(1)-17) Then Goto 50

    'SOSTITUZIONE NOTE
    14 If X(I)>X(I-1) Then X(I)=X(I)+1 : Locate 22,25 :
Pen 10 : Print "SOSTITUZIONE CON NOTA AUMENTATA" : Pen 4
: Locate 54,25 : Print N$(X(I)) : Goto 13
    If X(I)<X(I-1) Then X(I)=X(I)-1 : Locate 22,25 : Pen
10 : Print "SOSTITUZIONE CON NOTA DIMINUITA" : Pen 4 :
Locate 54,25 : Print N$(X(I)) : Goto 13
    50 Volume 50 : Play X(I)+K,0 : Rem suona la nota
corrispondente al tasto premuto
Loop
```

se risulta maggiore rispetto alla precedente; sarà minore in caso contrario.

Un esempio

Digitando il tasto "**A**" (cui corrisponde un **DO**) e, successivamente, premendo il tasto "**T**" (cui corrisponde la nota **FA#**, non facente parte della scala maggiore di DO) il programma, avendo riscontrato che FA# non fa parte della scala maggiore e che FA# risulta maggiore del DO dal punto di vista musicale, provvede a sostituire dapprima il FA# con il **SOL** (più alta rispetto al FA#) e poi, se il tasto "T" viene premuto una seconda volta, con il **FA** (più bassa rispetto al FA#).

Analogamente digitando il tasto "**G**" (cui corrisponde il **SOL**) e, successivamente, premendo il tasto "**E**" (che corrisponde alla nota **RE#**, non facente parte della scala maggiore di DO), il programma, avendo riscontrato che la nota RE# non fa parte della scala maggiore e che RE# risulta minore del SOL (sempre dal punto di vista musicale) provvede a sostituire la nota, ma stavolta dapprima con il **RE** (più bassa rispetto al RE#) e poi, se il tasto "E" viene premuto una seconda volta, con il **MI** (più alta rispetto al RE#).

Se, dunque, per errore, viene suonata una nota non facente parte della scala maggiore assegnata, il programma provvede a sostituirla con quella più alta o più bassa.

Può capitare che le note proposte per la sostituzione facciano parte dell'ottava *precedente* o di quella *successiva*. Se, ad esempio, consideriamo la scala maggiore di RE (tasto "**L**") e segniamo erroneamente un RE# (tasto "**P**") la nota potrà essere sostituita anche da un MI, che fa parte della scala maggiore di RE, ma che si trova nell'ottava successiva. Tale sostituzione può però avvenire solo quando il valore dell'ottava corrente è minore di **7**.

Analogamente, se per errore premiamo, sempre rimanendo nella scala maggiore di RE, il tasto "**A**" (che corrisponde a **DO**) la nota potrà essere sostituita con **SI**, che fa parte della scala maggiore di RE ma che si trova nell'ottava precedente.

Ecco spiegato il motivo per cui il vettore **N$** ha **18** elementi (numerati da **0** a **17**) invece di 16.

di Luigi Callegari

I modem US Robotics

Alta velocità, grande affidabilità, enorme risparmio in termini di scatti SIP; e prezzo adeguato...

Un modem è basilarmente un apparecchio che consente di *trasmettere e ricevere dati* tramite normali linee telefoniche. Si collega, oltre che ad una presa di alimentazione, *al computer* tramite un normale cavo RS-232 sull'interfaccia seriale ed *alla linea telefonica* tramite un normale cavo bifilare, con uno spinotto adatto alla spina installata dalla SIP.

Al modem può generalmente essere collegato *anche* l'apparecchio telefonico, in modo da poterlo utilizzare quando non si effettuano scambi di dati, ma si desiderano usare le proprie corde vocali, senza dovere collegare e scollegare modem e telefono.

Mentre per il cavo telefonico e la relativa presa non sussistono difficoltà, il cavo di collegamento alla porta seriale può essere causa di problemi. Infatti, sul mercato, si trovano molti cavi non adatti, o perchè hanno prese "maschio" e "femmina" che non si accoppiano ai corrispondenti connettori del computer o del modem (maschio-maschio o femmina-femmina), o perchè non hanno tutti i piedini collegati (spesso i modem, o meglio, il software di gestione richiede la presenza di collegamenti tra modem e computer tralasciati dai fabbricanti dei cavi) oppure ancora perchè li hanno nell'ordine sbagliato.

Lo **standard RS-232**, infatti, è un po' confuso e spesso si può fare confusione tra **cavi di prolunga** e **cavi per stampante seriale** che presentano, appunto, collegamenti *parziali* o non corretti per un modem.

In generale, per collegare con successo un modem al computer, occorre un cavo di prolunga, ovvero con **tutti** i piedini collegati *pin to pin*, come si dice in gergo. Ciò significa che il piedino **1** di uno dei due connettori del cavo deve essere collegato col piedino **1** dell'altro connettore, il numero **2** col numero **2** e così via.

Il software

Per fare funzionare un modem occorre anche(!) del software adeguato. In circolazione esistono molti programmi di pubblico dominio o commerciali, sia per **Amiga** che per **Ms-Dos**.

Tra i migliori ricordiamo **A-Talk III** (commerciale), **AZ-Comm** (shareware), **JRComm** (shareware) e **N-Comm** (shareware) per Amiga e **Qmodem** (shareware) e **Telix** (shareware) per **Ms-Dos**.

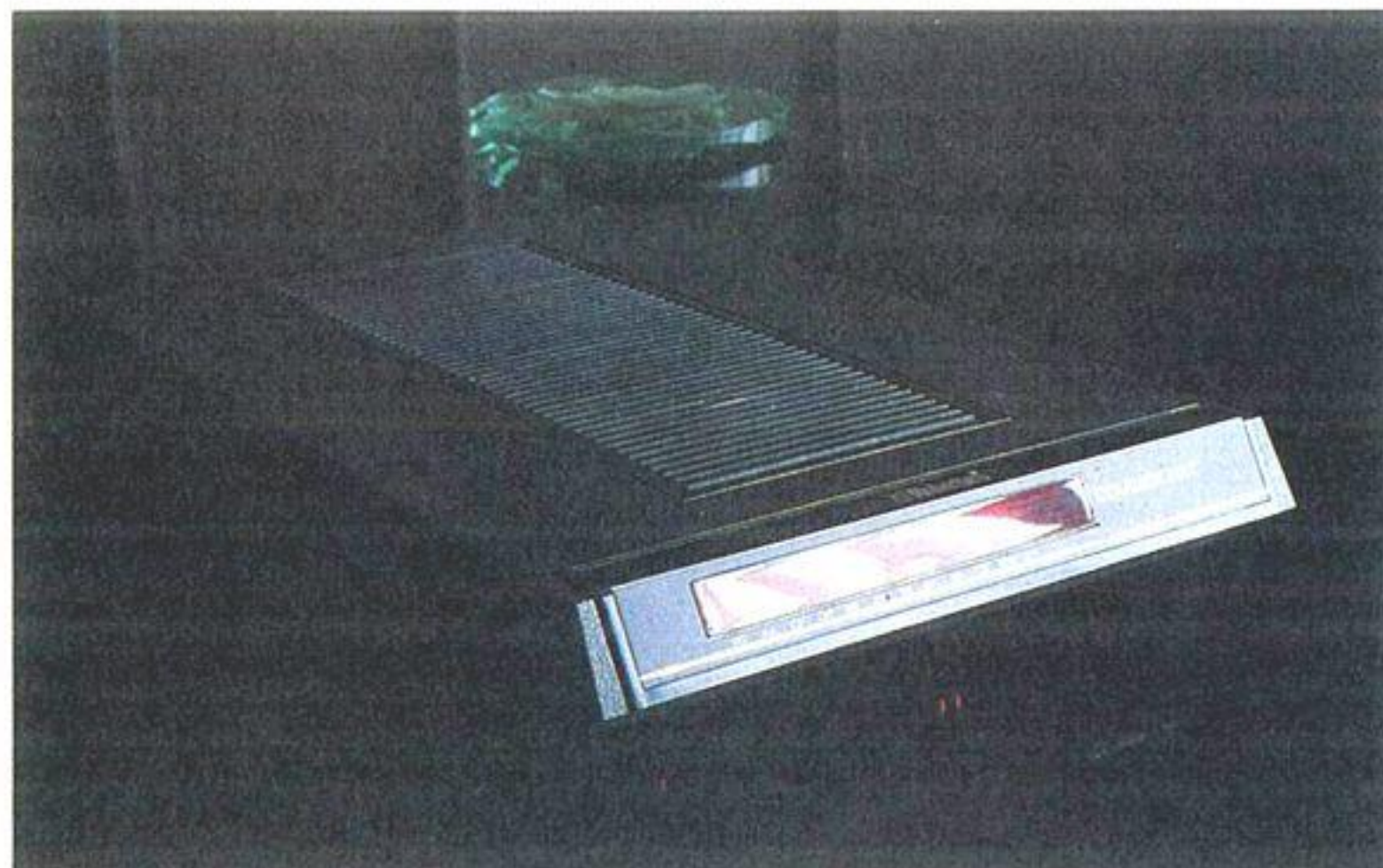
Un buon programma di comunicazione deve prevedere la possibilità di usare vari protocolli di comunicazione (almeno **ZModem** e **XModem**, di base), di variare i formati di comunicazione (il più usato è comunque lo standard **8N1**, ovvero pacchetti composti da **8** bit di dati, **nessun** bit di parità ed **un** bit di stop) e di eseguire agevolmente upload e download (invii e ricezioni di file) da banche dati. Ritorneremo in futuro su questi argomenti con articoli specifici, mentre i programmi di comunicazione per Amiga di pubblico dominio saranno inseriti sui prossimi numeri di **AmiGazzetta**.

I modem

Sino a pochi anni fa i modem in circolazione erano a 300 baud, ovvero consentivano di ricevere ed inviare dati a 300 bit al secondo. In seguito si passò ai 1200 baud, poi ai 2400 per arrivare ai recentissimi 9600 e HST.

In effetti, molti pensano che con queste velocità siamo vicini ai massimi fisici di trasmissione lungo i cavi telefonici della SIP, soprattutto se si considera che si tratta di **cavi non digitali** nella gran parte di Italia.

L'evoluzione dei modem, dovuta alle case produttrici, ha introdotto anche par-

ticolari sistemi software/hardware per **correggere gli errori** in trasmissione automaticamente da modem (senza l'intervento del software di gestione che gira sul computer) e **comprimere** i dati per aumentare la velocità ed il rendimento delle trasmissioni (e ridurre i costi dei collegamenti), altro argomento sul quale ci ripromettiamo di parlare in una futura serie di articoli su CCC.

La serie Courier

La **US Robotics** è una società americana nata costruendo modem per Apple, Zenith ed altre grosse case produttrici di elaboratori elettronici, conta circa 300 dipendenti, dei quali poco meno di un terzo sono ricercatori impegnati nello sviluppo della evoluta e speciale tecnologia necessaria per assemblare modem.

Nel 1987 la USR brevettò lo **standard HST**, che consiste in un particolare protocollo personalizzato per la trasmissione asincrona di dati in modo duplex asimmetrico. In parole povere, il modem invia e riceve dati allo stesso tempo lungo la linea telefonica, ma a velocità diverse, dato che su di un canale viaggia a **14.400** baud e sull'altro a **450**.

L'uso di questo standard, presente **solo** sui modem US Robotics, aumenta enormemente l'efficienza della trasmissione dei dati, grazie anche ai sistemi di **CRC** (**C**ycle **R**edundancy **C**heck), ovvero a tecniche hardware/software che consentono ai modem di rilevare e correggere in modo trasparente al terminale connesso (il computer) errori di trasmissione dovuti, tipicamente, a linee telefoniche disturbate.

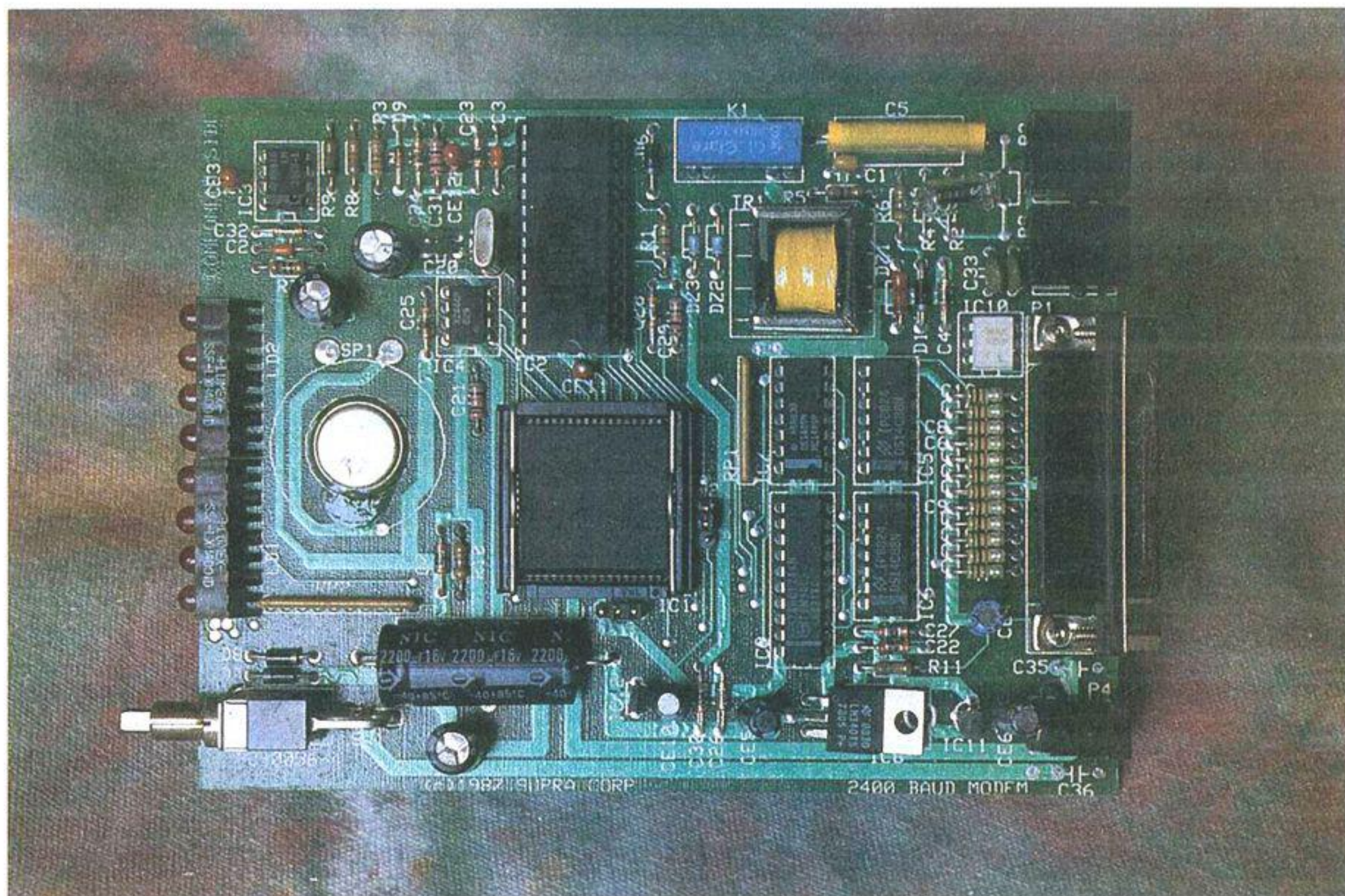
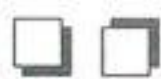
Tra le tante sofisticazioni dei modem della USR (come di altri modelli, in verità) troviamo la capacità di rilevare automaticamente la velocità di trasmissione del modem a cui sono collegati via telefono (possono passare dal colloquio a 14.400 baud a 300 nel giro di pochi istanti) e di attivare o disattivare automaticamente le varie tecniche di compressione di dati e di correzione di errori (**MNP, ARQ, V42bis**), sempre in funzione del modem con il quale dialogano.

Tutte queste regolazioni avvengono automaticamente tra i modem durante i primi istanti del collegamento, in modo del tutto trasparente all'operatore (o meglio, al terminale col programma di comunicazione), che deve soltanto avere regolato opportunamente i parametri di funzionamento del modem tramite gli appositi comandi **AT**.

Da prove sperimentali, risulta che la velocità massima effettiva di trasferimento dati dei modem USR si aggira intorno ai **35.500 baud** (bit al secondo) quando si stanno inviando file di testo, nel qual caso si può trasferire qualcosa come **500K** circa di testo ASCII in circa **due minuti**.

I modem Courier

La serie di modem più famosa della US Robotics è certamente la Courier, di costo superiore alla media dei prodotti concorrenti, ma con caratteristiche tec-

nologiche decisamente all'avanguardia. Tra l'altro, i modem della linea Courier sono perfettamente compatibili con tutti gli standard internazionali (**Bell** e **CCITT**), usano **MNP** di livello da **1** a **5** e compressione di dati **V42 bis**. Si noti che sopra i 2400 baud vi sono attualmente solo due standard accettati internazionalmente dalla CCITT: il V32 (9600 baud) ed il recentissimo V32bis (14.400 baud).

Attualmente la linea Courier consiste di tre modem al alta velocità: il Courier **HST**, il Courier **V32** ed il Courier HST **Dual Standard**. I primi due sono specializzati nell'uso del protocollo HST e V32, rispetttivamente, mentre il terzo modello, quello di punta, racchiude le caratteristiche dei due modelli precedenti (ed anche qualcosa di più).

Courier HST

Il modem Courier HST è disponibile sia su **scheda** che come apparecchio **esterno**.

Di regola, potendo spendere qualche lira in più e non avendo problemi di spazio sulla scrivania di lavoro, è preferibile acquistare il modello esterno per varie ragioni: è più facile trasferirlo da un computer all'altro (tra Ms-Dos e Amiga, ad esempio) funziona con qualunque tipo di computer dotato di interfaccia seriale (mentre le schede sono solo per Ms-Dos *oppure* per Amiga) e consente di controllare tramite i LED frontali che cosa sta accadendo.

Il Courier HST può funzionare in modo HST sino a **14.400 baud**, ma soltanto se collegato con un modem che supporti questo standard, ovvero un Dual Standard od un altro HST. Altrimenti la massima velocità utilizzata è di **2400 baud** (con correzione di errori e compressione dati MNP e/o V42 bis).

La velocità di dialogo viene regolata automaticamente dal modem in funzione della qualità del segnale della linea telefonica e del modello di modem presente, e riconosciuto, all'altro capo del cavo telefonico.

Courier V32

Questo modem **non** è un HST, quindi non consente la velocità di 14.400 baud ma al massimo la **V32**, ovvero **9600 baud** bidirezionali.

Esistono particolari programmi che consentono di usare questa caratteristica per ricevere ed inviare file contemporaneamente a tale velocità. Ricordiamo che lo standard HST, invece, può dialogare **soltanto** a 14.400 in una direzione, dal momento che l'altro canale è riservato al modem per eseguire le correzioni di dati.

Essendo pienamente compatibile con lo standard CCITT sino al V32, questo modem può funzionare a **qualunque velocità** di trasmissione tra 300 e 9600 baud con qualunque altro modello di modem standard CCITT.

Grazie al massimo livello di correzione d'errore ed al V42 (oltre alla compressione) si dovrebbe teoricamente riuscire a ragiungere una velocità di trasmissione di **38.400** bit al secondo. In effetti, da prove pratiche fatte con un file di testo, si può verificare che la velocità massima effettiva si aggira intorno ai 24.000 baud.

Courier HST Dual Standard

Il modello di punta della linea Courier, recentemente introdotto, oltre a tutte le velocità standard (300 / 9600 bidirezionali) ed a quella esclusiva della US Robotics (HST con 14.400 / 450 baud) offre anche il **V32 bis**, che consente una velocità massima di dialogo di 14.400 baud bidirezionalmente, raggiungendo una velocità di trasferimento pari ai 35.500 baud con file di testo, raddoppiati materialmente se si usa un programma che consente di inviare e ricevere dati contemporaneamente. Tale modem è in grado di riconoscere automaticamente l'apparecchio collegato all'altro capo della linea telefonica nei primi istanti (durante lo "scrambling") della connessione e di predisporsi (se precedentemente autorizzato a farlo dall'operatore tramite appositi comandi) alla velocità e con i protocolli più adatti.

Conclusioni

I modem della US Robotics sono uno standard qualitativo riconosciuto a livello mondiale.

Non sono forse i modem più diffusi, dato il prezzo, ma lo sono sicuramente in quei campi di applicazione (banche dati, reti di terminali) dove sono indispensabili qualità ed affidabilità, accanto ad una tecnologia all'avanguardia.

Lo standard **HST** è utilizzabile solo con modem della **USR**. In alcuni ambienti, come ad esempio quello delle banche dati amatoriali (**Fidonet**) e non, si tratta di uno standard *de facto* ben consolidato, che consente alta efficienza di lavoro e, comunque, l'uso di tutti gli standard CCITT e Bell sino a 2400 baud.

Il modello V32 è particolarmente appetibile, invece, se si considera che si prevede una pioggia di modelli "taiwanesi" dotati di questo standard che, non costituendo brevetto di una sola casa, avrà presumibilmente una grossa diffusione per il gran numero di pezzi prodotti a bassissimo costo.

Il modello Dual Standard è un piccolo gioiello tecnologico, indispensabile per le banche dati, consentendo di lavorare con la massima velocità ed efficienza indipendentemente dal modello di modem a cui si collega.

di Luigi Callegari

Professional Page, Desk Top Publishing... italiano

Anche i programmi di impaginazione, finalmente, iniziano ad essere commercializzati con manuali tradotti in italiano

Il mondo del **Desk Top Publishing**, ovvero dell'impaginazione di documenti tramite personal computer, è in continua espansione ed evoluzione.

Si tratta di un campo di applicazione, infatti, che può interessare moltissimi utenti potenziali od effettivi: dallo studio di un professionista che deve produrre documenti ben composti od efficaci, allo studente che deve redigere una ricerca per la scuola.

Amiga è un'ottima macchina per effettuare impaginazione, in quanto alle caratteristiche hardware evolute (che tutti conosciamo) affianca alcuni pacchetti che hanno poco o nulla da invidiare a quelli presenti in altri sistemi dalla nomea "professionale", come gli Apple Macintosh o gli Ms - Dos.

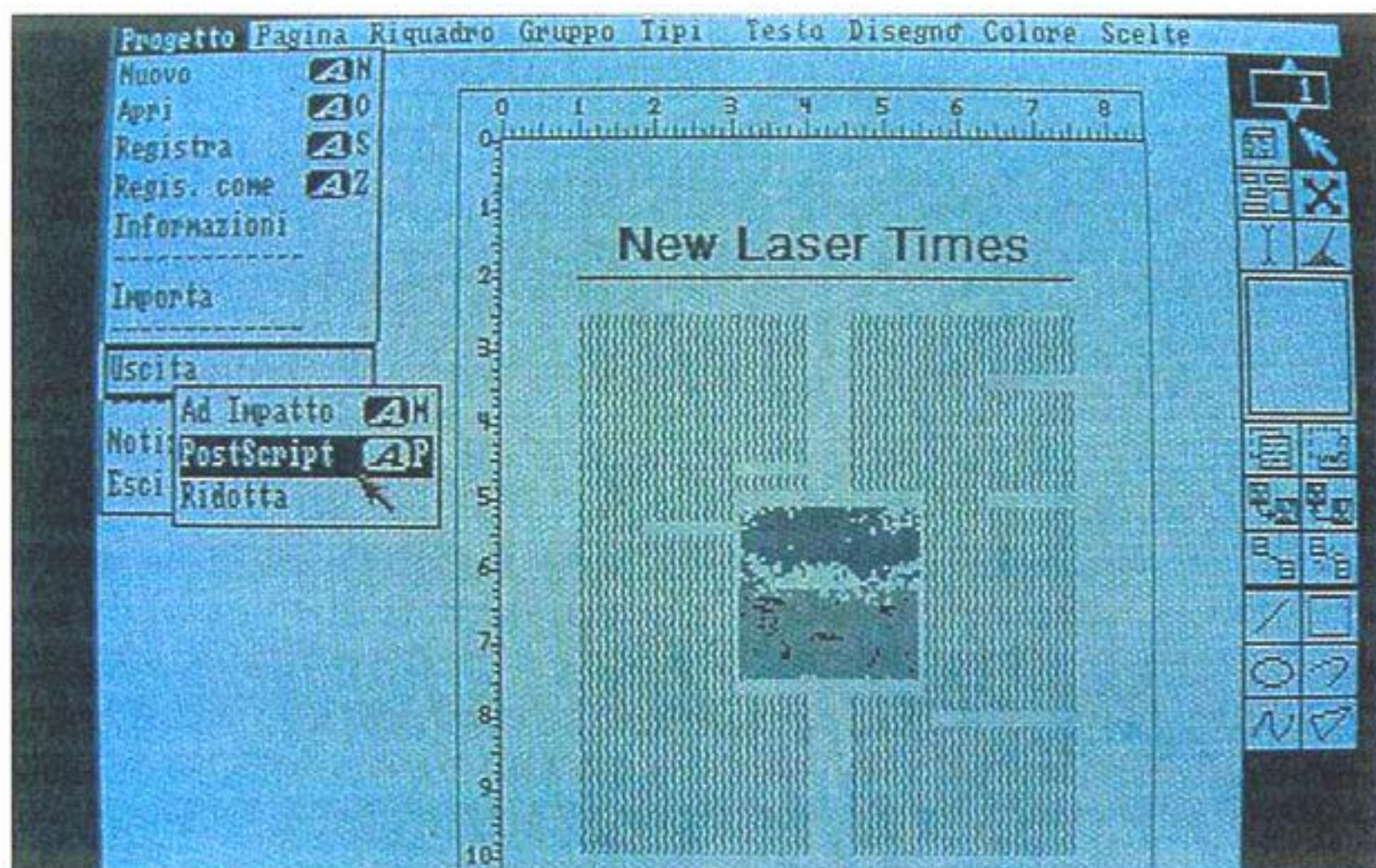
Recentemente la **Leader** ha effettuato la traduzione dei manuali e del programma **Professional Page**, uno dei più diffusi per Amiga, utilizzato in Europa ed all'estero per realizzare alcune riviste dedicate, interamente Amiga, senza l'ausilio di nessun altro tipo di computer.

Installazione

Professional Page **V1.3** richiede almeno **1Mb di RAM** ed un floppy disk drive per funzionare, ma la *configurazione ideale* per un utilizzo pratico ed efficiente consiste in un Amiga 2000 con scheda VDE (Video Display Enhancer, contro il flickering) oppure un Amiga 3000 e monitor ad alta risoluzione, dotati di 3Mb di RAM almeno e di un disco rigido.

E' utile disporre anche di 1Mb di CHIP RAM, come consentito dai nuovi chip custom, perchè diviene così possibile fare girare altri programmi contemporaneamente senza incorrere in situazioni pericolose: Professional Page, infatti, richiede almeno 720K di RAM, dei quali 360K di Chip Ram, per funzionare, ma tale quantità aumenta vivacemente con l'uso di particolari funzioni.

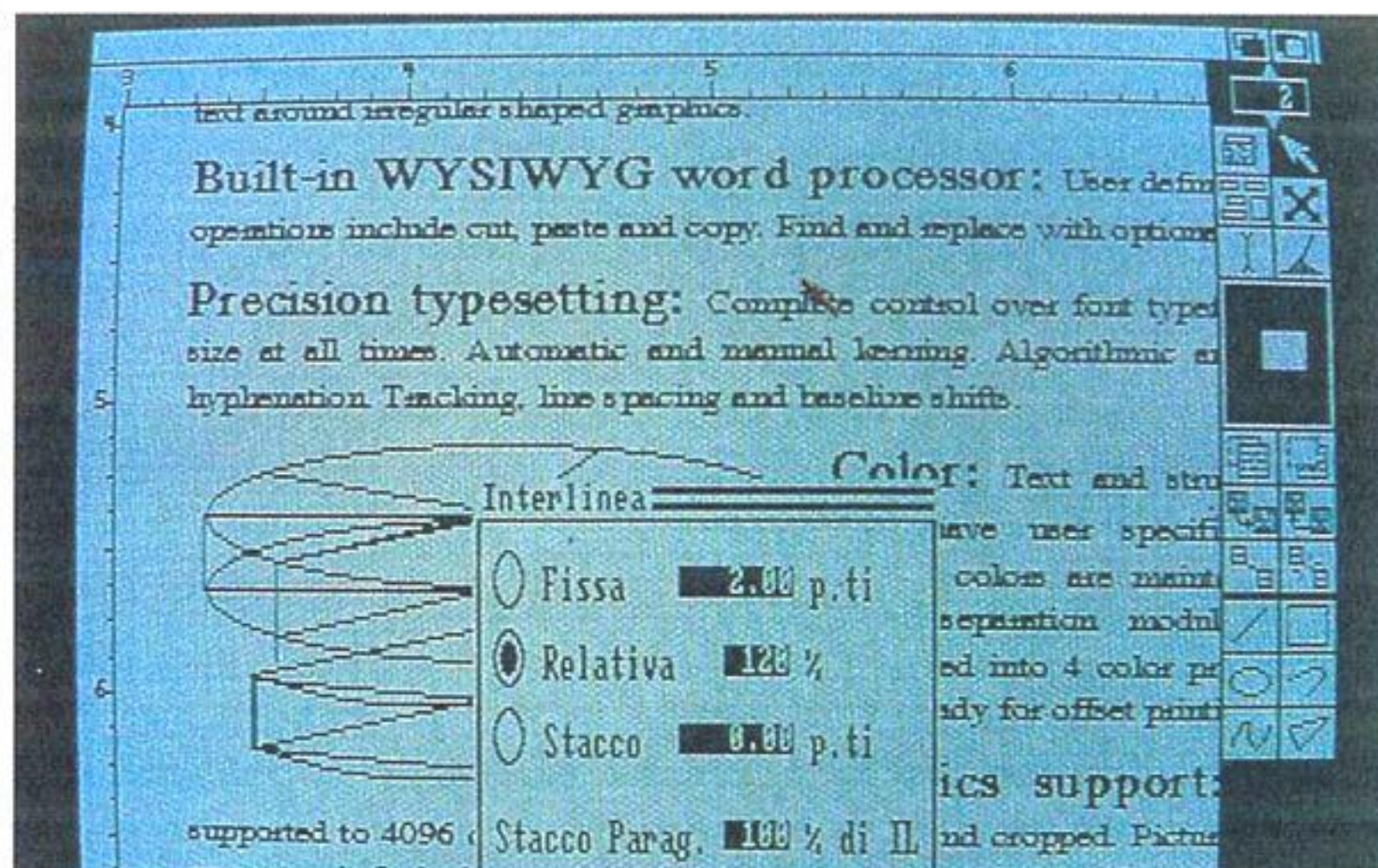
Per un uso professionale del prodotto è vivamente consigliabile, per non dire indispensabile, possedere un computer veloce, quindi un Amiga 3000 (con 68030 a 16 oppure 25 Mhz) od un Amiga 2000 dotato di scheda acceleratrice (GVP o A2630). In questo modo, a fronte di una spesa supplementare, si ottiene una velocità operativa ragionevole e decisamente superiore, ad esempio, a quella visibile su sistemi **Macintosh** "base" o sui normali Ms - Dos con processori 80286 o 80386 a 16/20/25 Mhz.

Per installare il programma su disco rigido è fornito sul disco **Fonts e Utilities** un apposito programma lanciabile anche da Workbench che, una volta specificata la directory del disco rigido che si desidera usare, vi trasferisce tutti i file necessari al funzionamento.

Uso pratico

Alla partenza del programma compare un'area di lavoro praticamente "standard" tra i programmi di impaginazione. Occhi esperti possono però notare le funzioni di gestione delle fonti di carattere a video e su stampanti non PostScript.

Nelle opzioni di configurazione dell'ambiente di lavoro, sotto il menu **ToolTypes**, troviamo le regolazioni inerenti il tipo di schermo (Workbench o personalizzato del programma, nelle varie risoluzioni grafiche e cromatiche di Amiga), unità di **misura** (pollici, centimetri e pica), **formato** del foglio (standard USA, legale USA, A3, A4, A5, B5), **sillabazione** (in lingua inglese, francese, spagnola od italiana) e scelta della porta su cui inviare l'uscita **PostScript** diretta ad una **stampante** (parallela o seriale).

Durante la creazione di nuove pagine, sia quella di base (opzione **From Default**) sia di quelle mastro (**From Template**), l'utente ha la possibilità di ottenere direttamente le strutture per il testo in corrispondenza delle colonne, nonchè di averle collegate in sequenza tra loro per facilitare lo scorrimento del testo al loro interno (opzione **Automatically link columns**).

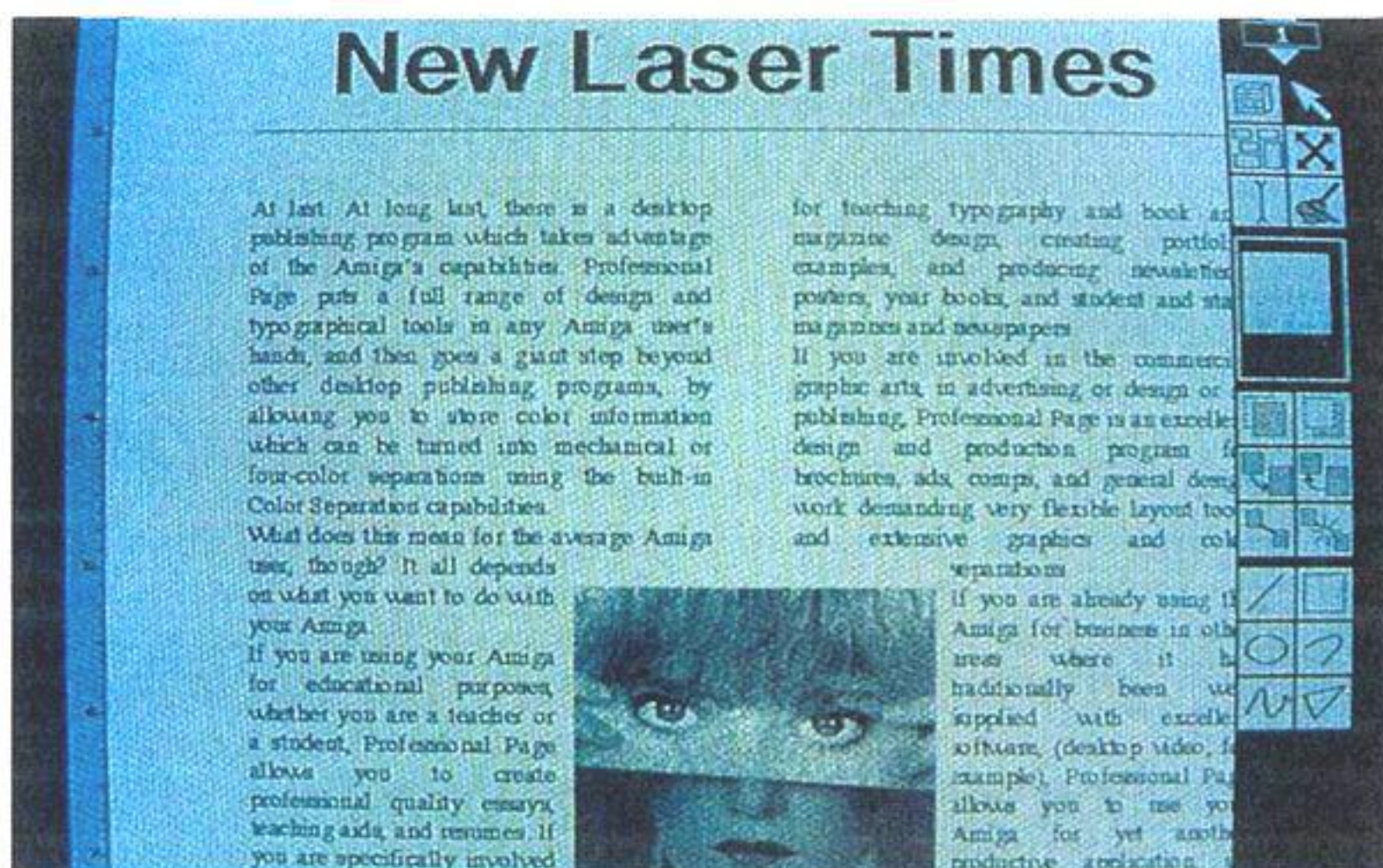
Creando una nuova pagina, è possibile decidere quali delle strutture appartenenti ad una pagina creata su modello di quella di mastro, devono essere o meno "bloccate" (cioè non modificabili per quanto rigurda le dimensioni e la posizione prima di togliere tale blocco): se si vuole disabilitare questa opzione basta agire sulla finestra di dialogo apposita disattivando l'opzione **Lock Boxes**.

Le dimensioni massime di una pagina sono di 22 x 22 pollici, cioè circa 55 x 55 centimetri, in funzione del modello di stampante disponibile.

Immagini e testo

Professional Page supporta l'importazione di testo creato con alcuni dei **word processor** più diffusi per Amiga: **Scribble, Word Perfect, Excellence e Transcript**. In effetti, Professional Page consente di usare quest'ultimo programma, prodotto dalla Gold Disk stessa, come "editor" di rifinitura diretta del testo *durante* l'impaginazione stessa.

Dato che Professional Page, come tutti i programmi di impaginazione, non dispone di tutte quelle funzioni e sofisticazioni proprie dei programmi di videoscrittura pura, ci sembra molto intelligente consentire di richiamare a piacere un efficiente word processor *durante* l'impaginazione per apportare modifiche significative al testo su cui si sta lavorando. Generalmente, bisogna invece *uscire* dal programma di impaginazione, *caricare* la videoscrittura, quindi il file di testo, *apportare* le modifiche, *uscire* dal programma di word processing e *ricaricare* l'impaginatore. Con Professional Page si può fare tutto restando all'interno del programma principale, avendo cura di mantenere a disposizione, su disco, il w/p Transcript.

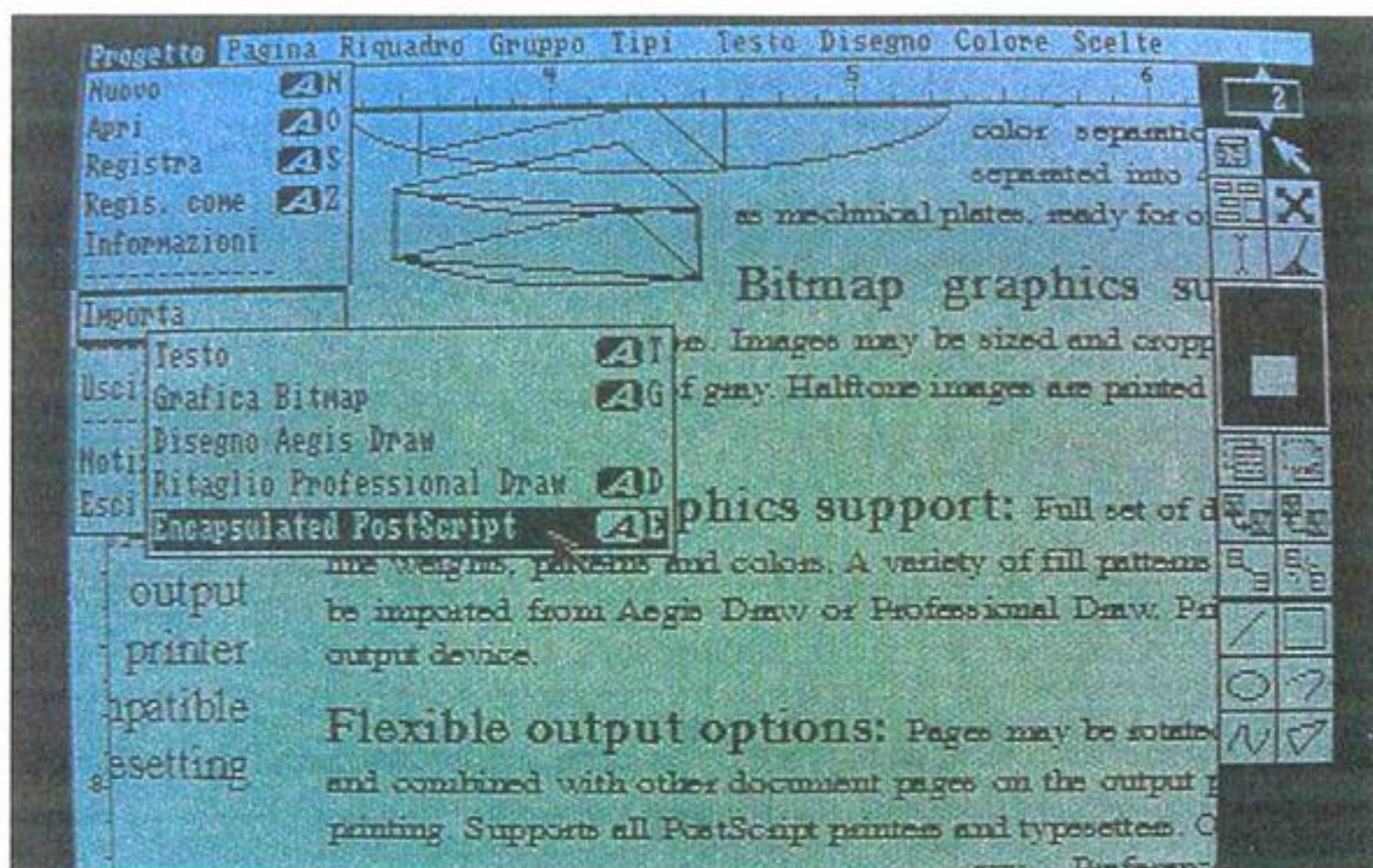
Alcune caratteristiche che abbiamo visto in altri programmi di DTP (come ad esempio **Saxon Publisher** e **PageStream**), ma che mancano in Professional Page, sono la possibilità di manipolare testi come oggetto (rotazioni, distorsioni, parole che seguono un percorso curvilineo, eccetera).

Un possibile rimedio, suggerito dal manuale, è di usare il programma di grafica vettoriale **Professional Draw** per creare gli effetti voluti, che sono poi importabili all'interno del testo tramite un'apposita opzione di Professional Page come **clips**. Altra possibilità, avendo una stampante PostScript, consiste nel potere

scegliere dall'apposito pannello di selezione della stampa (**Postscript Output Specs**) di ruotare il testo voluto di un certo angolo scegliendo anche la funzione **No Eject**, in modo che la stampante sovrapponga le due pagine contigue e la creazione dell'effetto di rotazione desiderato (procedimento un po' macchinoso...).

La grafica di tipo "bitmap", generata da programmi come **Deluxe Paint** o **Photon Paint**, può essere importata tranquillamente, memoria permettendo, senza limiti di formato massimo (sino al 1024 x 1024 massimo possibile per Amiga). Ciò consente di usare anche degli scanner monocromatici, quali quelli venduti dalla stessa Gold Disk, che raggiungono la risoluzione di 300 punti per pollice e quindi possono generare pagine grafiche bitmapped di ragguardevoli dimensioni.

Un'altra possibilità di lettura di file grafici deriva dalla funzione **Import Encapsulated Postscript**: in questo modo si possono caricare, nel documento in impaginazione, i moduli PostScript generati con un qualunque altro programma (anche su altri calcolatori: il formato PostScript è standardizzato perfettamente), come ad esempio i famosi **Illustrator** o **FreeHand** di Apple **Macintosh**.

Stampa

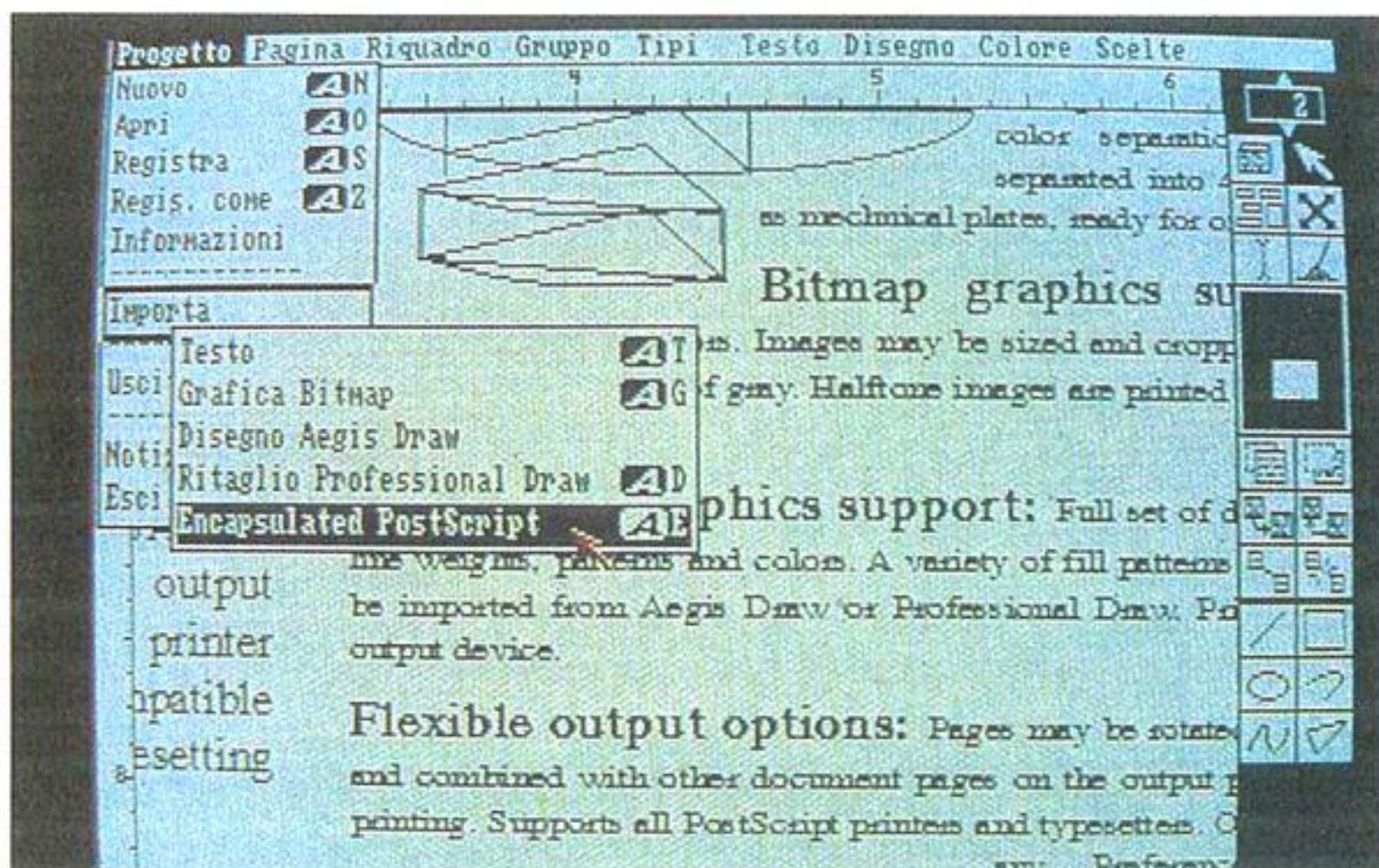
Un programma di impaginazione serve a ben poco se non consente di rendere su carta ciò che si è elaborato.

In Amiga, come negli altri calcolatori, i caratteri a video sono generati tramite delle mappe di bit (bitmap) prefissate in funzione dei vari stili, per velocizzare al massimo la loro visualizzazione. Le mappe dei bit che formano i caratteri sono contenute nelle subdirectory di Fonts del disco di sistema e caricate in memoria Ram, per l'elaborazione, solo quando necessarie.

Essendo impossibile creare le bitmap relative alle sagome di *tutti* i caratteri in *tutte* le possibili grandezze (per ovvi motivi di spazio) qualora l'utente richieda un formato non presente nella directory, Amiga provvede a calcolare la bitmap partendo da quella più *simile* già disponibile. Così si ottengono le ovvie seghettature dei caratteri, che offrono una immagine a video di qualità assai deludente. Il fenomeno è particolarmente evidente su computer come Amiga o Macintosh, dotati di una evoluta interfaccia grafica, ma non in quelli dotati di interfaccia "a caratteri" ormai sorpassati (persino Ms - Dos ora si sta convertendo all'interfaccia grafica con i vari Windows e GeoWorks...).

Per quanto riguarda le fonti di caratteri per stampante, comunque, la Gold Disk fornisce nel pacchetto di Professional Page delle fonti della Compugraphic (una società del gruppo Agfa Gevaert), limitati a due (**CSTimes** e **CSTriumvirate**, corrispondenti ai classici **Times** e **Helvetica**) ma acquistabili a parte.

I caratteri di queste fonti non sono di tipo bitmapped, ma vengono calcolati in base ad una descrizione delle loro caratteristiche (in gergo si dice che sono generati "algoritmicamente") che consente di ottenere sagome di qualunque dimensione senza seghettature sia sul video che su stampanti non PostScript, utilizzate in modo grafico.

Conclusioni

Profesional Page **V1.0** è stato il primo programma di DTP professionale per Amiga. La versione è tradotta con molta cura e competenza sia per quanto riguarda il manuale, sia per quanto riguarda il programma; rappresenta una evoluzione significativa in attesa della versione 2.0 attesa a brevissimo tempo.

di Gregor Samsa

Amiga Amos, ed il Basic diventa Superlinguaggio

AmigaBasic (diciamo la verità) è lento, scomodo, limitato; ma per un principiante, per fortuna...

Propiziato dalla ormai leggendaria inefficienza e scomodità dell'interprete MicroSoft fornito in dotazione al computer, si è assistito negli ultimi tempi ad un proliferare di nuovi **Basic**, tesi soprattutto a sfruttare al meglio l'ambiente Amiga.

Ambiente ricco di risorse fin troppo succulente per un aspirante programmatore, che, a meno di ricorrere a linguaggi come il **C** oppure l'**Assembler**, si ritrova quasi sempre a doverle ignorare o emulare senza mai potersi avvicinare a risultati di rilievo.

Eppure il Basic, nonostante sia tradizionalmente il linguaggio per "chi comincia", ha in genere raggiunto livelli di strutturazione ed efficienza tali da reggere il confronto tranquillamente con altri più blasonati cugini di alto livello (tipo il **Pascal**, per intenderci).

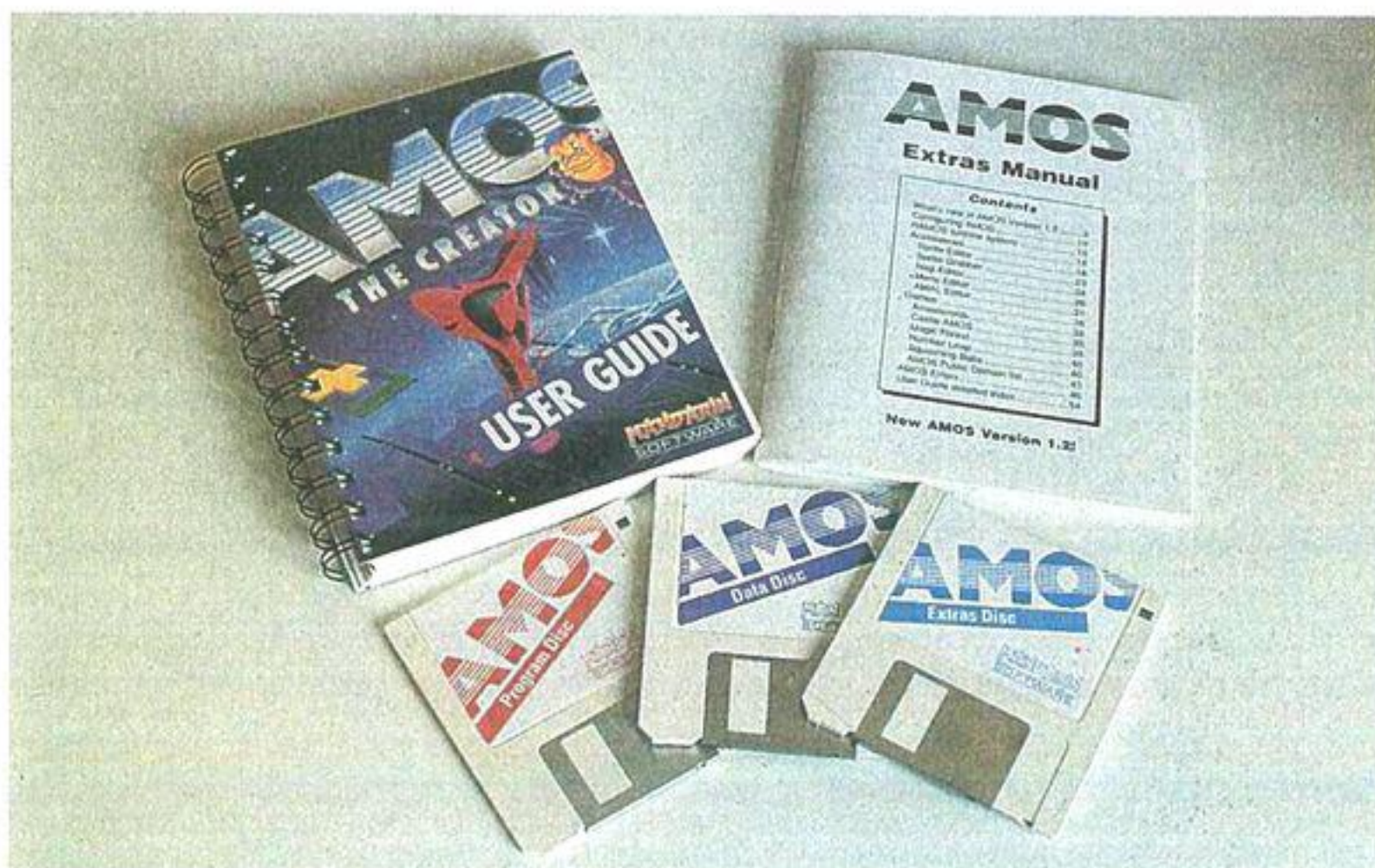
Amos, giunto ormai alla versione **1.2**, è, senza mezzi termini, uno di quei prodotti in grado di far tornare il sorriso a qualunque estimatore del Basic, e soprattutto a qualunque estimatore di Amiga. Pur mantenendo la maggior parte degli aspetti formali del linguaggio, ed anzi arricchendoli ulteriormente, questo interprete della **Mandarin Software** raggiunge livelli di sofisticazione e velocità tali da avvicinarsi concretamente a prestazioni da "assemblisti". Se a ciò si aggiunge che dovrebbe essere in arrivo (se non già in circolazione quando leggerete queste righe) un **compilatore Amos-dedicato**, non ci vuol molto per presagire una diffusione pressoché totale del prodotto, in barba a qualunque AmigaBasic.

Pur non tentando nemmeno, in questa sede, una descrizione approfondita (peraltro impossibile in poche pagine) e con la **promessa di approfondirne la conoscenza in una rubrica specifica** (sempre che le richieste non manchino...), vediamo di delinearne le caratteristiche essenziali.

L'ambiente

Già dal primo impatto, e soprattutto se abituati a certe velocità di AmigaaBsic, ci si rende conto della filosofia di Amos.

L'interprete prende in mano completamente le risorse di Amiga, creando un proprio ambiente esclusivo. Tanto esclusivo che, anche se si è lanciato Amos dal Workbench di un proprio dischetto, una eventuale regolazione della **tastiera** italiana non risulterà più valida nell'ambito dell'editor.

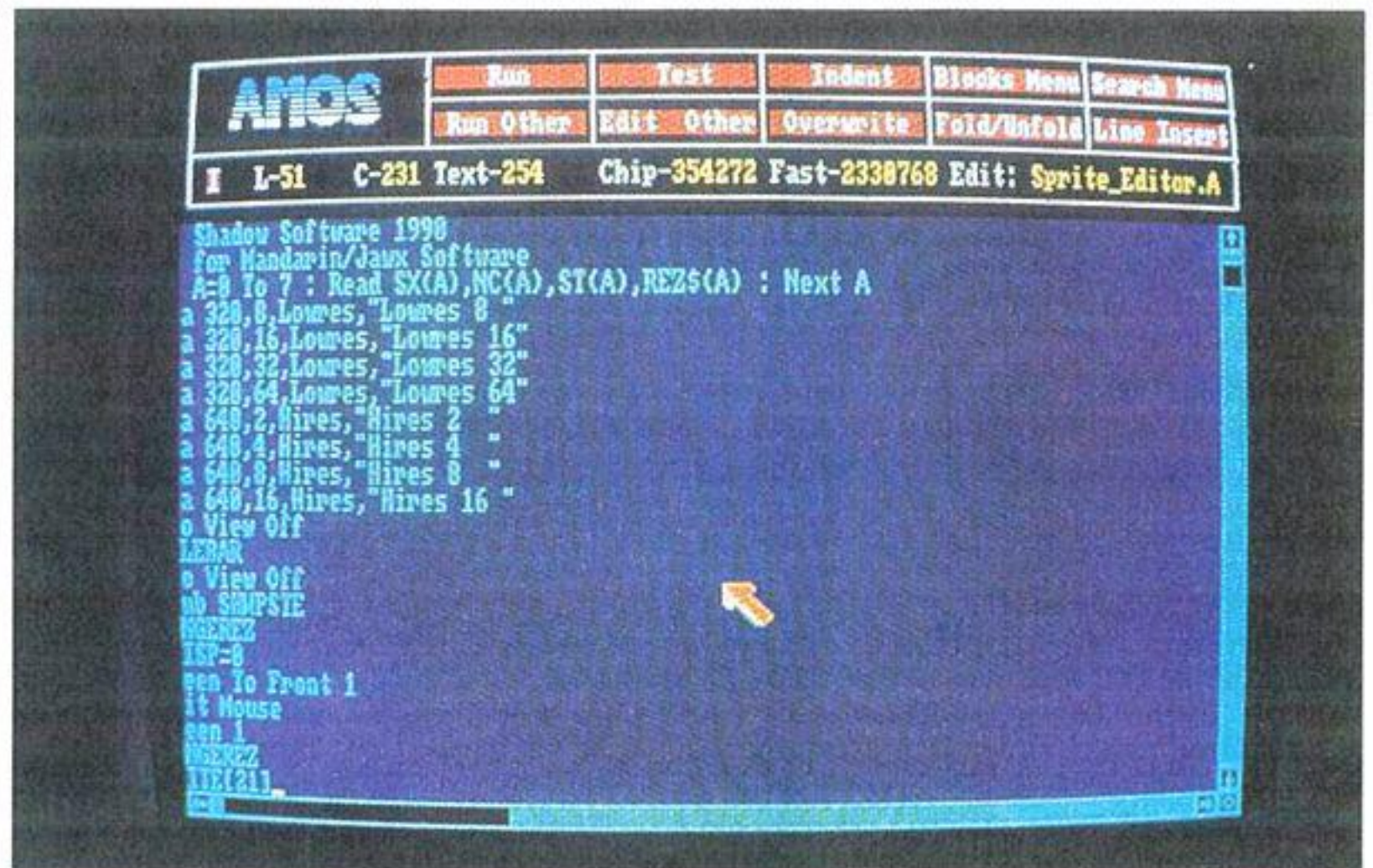
Niente di complicato, per carità; solo che l'interprete gestisce una sua mappa di tastiera che va adeguata di conseguenza, se necessario. E se non si trovasse quella adatta, nessun problema: basterebbe lanciare un programma scritto in Amos Basic presente nel disco principale (**Keyboard_Definer**) per costruirsi, a semplici colpi di mouse, una mappa su misura. La stessa filosofia, per inciso, Amos la adotta anche in settori molto più proficui, come ad esempio il **multitasking**. Questo, a livello Amiga, ovvero

all'*esterno* dell'editor, viene fortemente limitato, ma in compenso si dispone di un multitasking per così dire *interno*, in grado, per esempio, di consentire il caricamento di più programmi, e di eseguirli autonomamente uno dall'altro (con **Prun**), senza perdere quello al quale si sta lavorando.

Ma ci si sta già addentrando in argomenti particolari, per cui torniamo a dare uno sguardo a quanto viene proposto subito dopo il lancio di Amos, l'editor vero e proprio.

Lo spazio di lavoro occupa l'intero schermo, mentre sulla parte alta sono presenti numerosi riquadri a formare un vasto menu di scelta, adoperabile con il consueto click del mouse.

Consueto... ma non troppo!

Provando, infatti, a pressare il pulsante **destro** del mouse mentre il puntatore si trova nella zona menu dell'editor, ci si accorge che le voci contenute nei riquadri vengono sostituite da altre, per altrettante scelte. Lo stesso effetto lo si raggiunge adoperando lo **Shift**, ma non è ancora finita: altri set di opzioni sono raggiungibili (sempre negli stessi riquadri) adoperando i tasti **Ctrl**, **Alt**, Amiga **sinistro** ed Amiga **destro**, per un totale di **60 opzioni** (i due tasti Amiga accedono, ognuno, ad un set diverso).

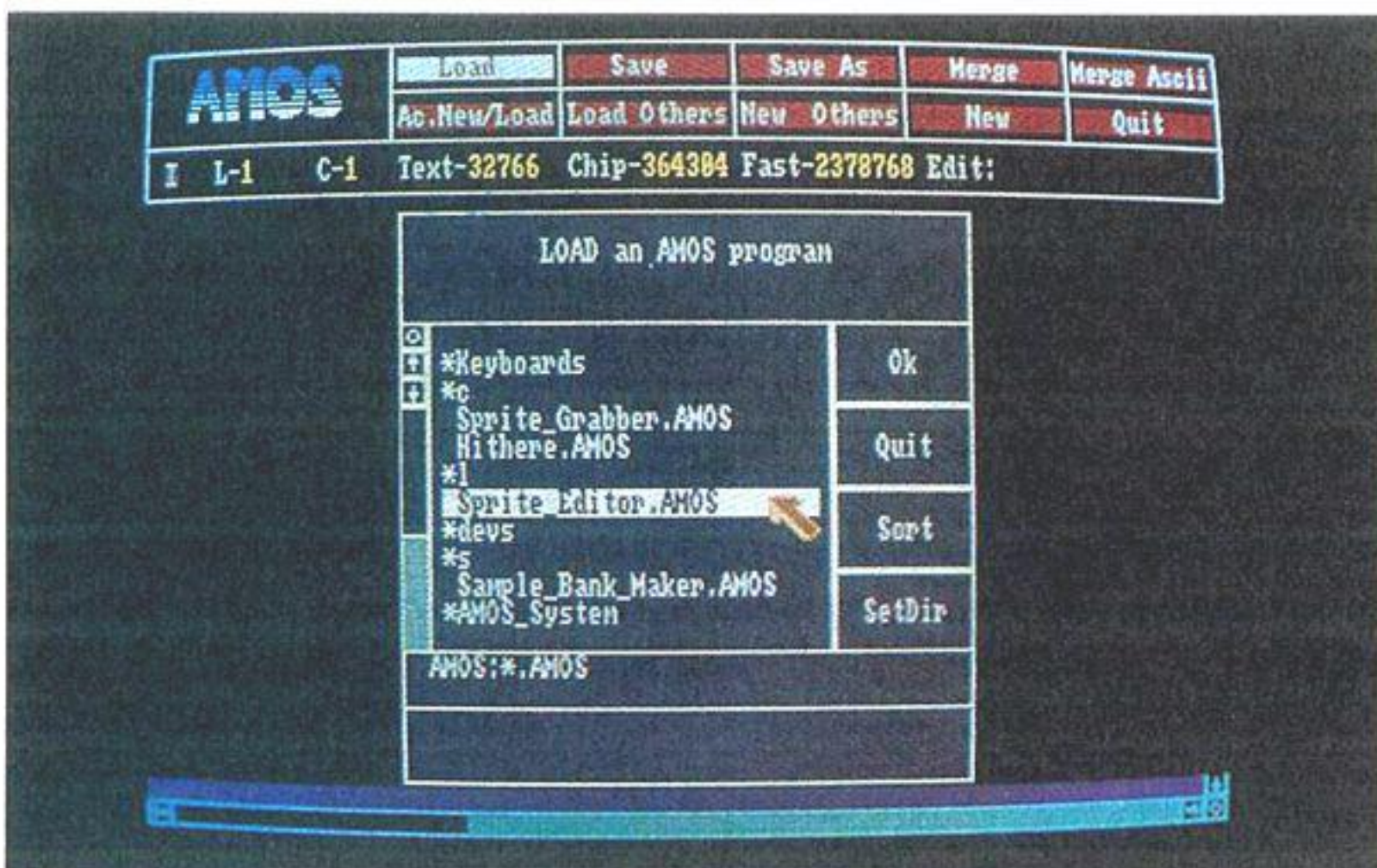
Oltre che con il mouse, le varie scelte sono attive anche sui **tasti funzione**, premuti da soli o in associazione con i suddetti tasti di controllo. A condire il tutto, le scelte legate ai tasti Amiga sono **riprogammabili**, e possono essere sostituite da vere e proprie **macro** con il comando **Key$**, tanto in modo diretto che nell'ambito di un programma.

Subito sotto i riquadri che fungono da menu, è visualizzata una **riga di stato** che riporta riga e colonna nella quale si trova il cursore, la memoria libera di sistema (**chip** e **fast**), e le dimensioni del **buffer** riservato al testo del listato.

Lo schermo di lavoro è inoltre dotato di **slider** (barre di scorrimento della finestra) sia orizzontali che verticali, che consentono tanto uno scrolling fine che uno più macroscopico. Se il pensiero torna all'editor di Amigabasic, si rischia una crisi isterica, accentuata dalla presenza di un mini-**help** in linea, raggiungibile con la pressione (guarda caso) del tasto Help.

Inutile aggiungere che i movimenti del cursore sono fluidi e veloci, che sono consentiti spostamenti ad inizio e fine riga o da istruzione ad istruzione, e che esiste addirittura la possibilità (con **Alt** + cursori verticali) di far scorrere le etichette e le procedure presenti nel listato.

Oltre al *modo editing*, un interprete che si rispetti deve possedere un *modo diretto* di impartire i comandi. Con Amos, si accede allo schermo di output premendo il tasto **Escape** (idem per rientrare nell'editor), cosa che permette anche di fruire di una sezione nella quale impartire i comandi, dotata di un proprio **prompt**, spostabile e rimaneggiabile nelle dimensioni. Niente, insomma, è affidato al caso: tutto ciò che serve è sempre disponibile... ma con qualcosa in più.

L'accesso ai **dischi** per le varie operazioni di caricamento / salvataggio è facilitato da un **requester** molto intuitivo e semplice da usare, lo stesso che (udite, udite) può essere richiamato anche da programma con una sola istruzione (**Fsel$**).

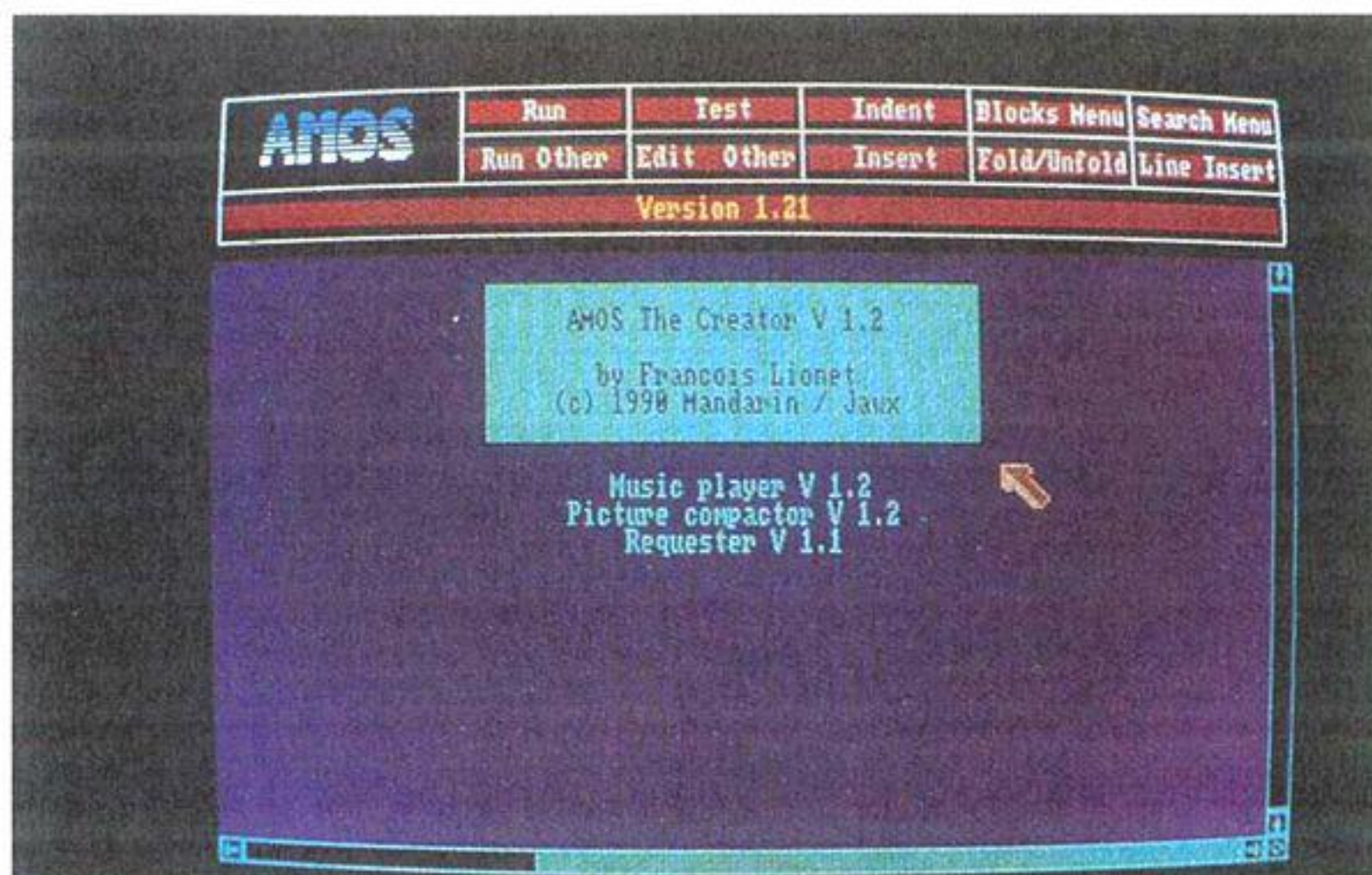
Il che significa, per esempio, che basterà impartire in modo diretto (o da programma) qualcosa come `Print Fsel$("df0:")` per vedere apparire il requester, e dopo aver selezionato col mouse il nome di un file, lo stesso nome verrà stampato sullo schermo. Con le implicazioni che si possono immaginare. Roba che, per ottenerla, nella migliore delle ipotesi occorreva prima scomodare

la Arp.library, e comunque a suon di C ed Assembler dei più tosti...

Il Basic

Sin dalla sua comparsa sul mercato, il nome Amos è subito stato associato alla creazione di videogames.

E non a torto: basta lanciare uno dei vari giochi memorizzati nei dischetti del package (con tanto di listato spulciabile) per non restare sbalorditi dalla fluidità delle animazioni, dal sofisticato sonoro, e da alcune manipolazioni che richiedono inevitabilmente un intervento quasi a livello hardware di Amiga. Tutto l'ideale per confezionare games di prima qualità, insomma, ma in effetti con Amos ci si può fare di tutto, magari privilegiando una presentazione grafica, o un interfacciamento ultraintuitivo.

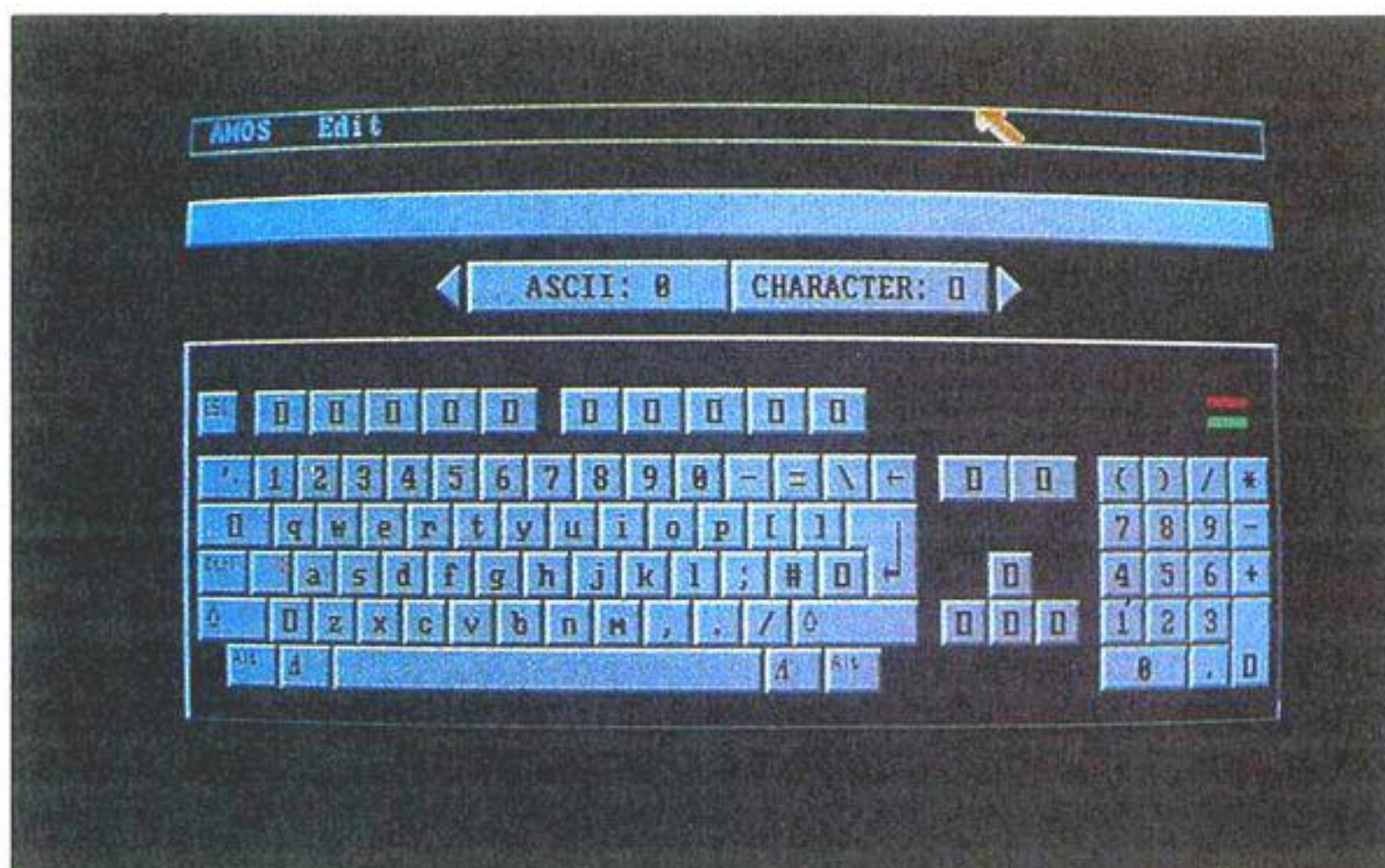
L'interprete dispone di **centinaia di comandi**, molto spesso varianti sullo stesso tema, che in generale consentono di sostituire con **una sola istruzione** l'uso di più **librerie di sistema**, o di lunghe routine per ottenere certi scopi (come il `Fsel$` prima visto). Considerata la vastità, non ci soffermeremo singolarmente su di esse, ma qualche accenno alle possibilità del linguaggio non guasta.

Intanto, per chi già mastica un po' di programmazione, va detto che Amos Basic si presta ad una notevole strutturazione, grazie alle sue **Procedure** (l'equivalente dei sottoprogrammi di Amiga Basic) di pascaliana memoria, e ad una notevole elasticità nelle strutture iterative: oltre ai sicuramente noti **For... Next** e **While... Wend**, sono infatti implementate anche le forme **Repeat... Until** e **Do... Loop**, a garantire maggiori possibilità di controllo. Le Procedure, come del resto in Amiga Basic, consentono l'uso di **variabili locali**, ma con Amos la scelta è più vasta: si possono far condividere queste variabili alla procedura ed al programma principale (il ben noto **Shared**), ma anche a tutte le altre procedure (**Global**).

Il ricorso ai numeri di riga è facoltativo... ma largamente sconsigliato: l'uso delle **etichette** risulta molto più pratico, soprattutto in forza della possibilità di rintracciarle immediatamente tramite le facilitazioni dell'editor.

Altre piccole ma gradevoli sorprese, prima di passare a quelle veramente importanti, sono rintracciabili nei settori più svariati. Qualche esempio: tempo addietro proprio sulla nostra rivista si sono pubblicate due routine che verificavano la presenza o meno di un file su disco. Con Amos, cose del genere diventano superflue: basta adoperare la funzione `Exists(file$)`, che restituisce l'informazione sotto forma di valore vero/falso. Sempre in rapporto al Dos, non sono poi da ignorare **Dfree**, che restituisce lo spazio libero su disco, e vari altri comandi che permettono i comuni **Rename**, **Makedir**, o altro. Il tutto, come ovvio, senza trascurare le normali capacità di accesso a files casuali o sequenziali, anche in questi casi con notevole facilità.

Si potrebbe ancora citare, tra la miscellanea, il più che semplice controllo sulla memoria disponibile, che può essere affidato a semplici Print **Chip Free** oppure Print **Fast Free**, ma lo spazio tiranno impone a questo punto che si accenni alle più potenti caratteristiche di Amos, quelle che riguardano la grafica.

Grafica

In pratica, si può fare di tutto. O almeno, tutto ciò che sembrava impensabile per un Basic. A cominciare, per esempio, dalla gestione delle schermate **IFF**. Certo ricorderete le lunghe e noiose routines

di acquisizione di questo formato in Amiga Basic, per non dire della loro inevitabile lentezza.

Ebbene, con Amos tutto ciò che occorre fare è impartire un banalissimo **Load Iff**, ed il gioco è fatto. Naturalmente esiste anche la possibilità inversa, ovvero salvare il contenuto dello schermo in formato Iff, anche se si trattasse di una schermata elaborata nell'ambito del programma. Il comando, in questo caso, sarebbe (indovinate un po'?) **Save Iff**.

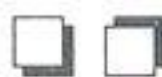
Ancora più interessanti, poi, i comandi di manipolazione degli **Screen**. Sempre con una sola istruzione, è possibile per esempio far scrollare lo schermo come più aggrada, o adoperare il fantastico **Dual Playfield**, che *fonde* due schermi in uno, con la possibilità di stabilirne priorità e movimenti, fino ad emulare perfettamente effetti simili a quanto si gode in Hang-on e similari.

Senza poterle elencare tutte, si pensi insomma a quanto comunemente sfruttabile (e non sempre, da Basic) tramite le librerie grafiche: per ogni funzione (e talvolta per più funzioni) esiste in pratica un equivalente comando Amos, che tra l'altro "gira" a velocità incredibile, se si pensa che ci si trova al cospetto di un interprete.

In più, sono implementati alcuni effetti di sicuro interesse: dalla rotazione ciclica dei colori (**Shift Up** e Shift **Down**) alla creazione di un arcobaleno (**Rainbow**) definibile. Per non dire di **Zoom**, che fa proprio ciò che ci si aspetta: ingrandisce una porzione di schermo!

Dove però si raggiunge un livello di eccellenza, anche se riservato ad una programmazione avanzata, è la possibilità di intervento sulle **Copper List**, per ottenere effetti grafici letteralmente impensabili per un interprete... ed in effetti non è più il Basic a funzionare, ma piuttosto una programmazione in linguaggio macchina "pilotata" da alcuni comandi messi a disposizione di Amos.

Come ovvio, non mancano comunque i più normali (ma facilitati) modi di accesso ad eventuali routine esterne (**Pload** e **Call**), né le tradizionali **Poke** e **Peek** per i vari formati **byte**, **word** e **longword**, ma anche in questo settore Amos mette qualcosa in più (la chicca): giusto per gli smanettoni più evoluti, c'è la possibilità di settare direttamente da programma i **registri** del microprocessore, o di testare singoli **bit**!

Altro che basic...

Altre feature

Ci sarebbe ancora da accennare al nutrito numero di istruzioni dedicate all'**animazione**, anch'essa resa di facilissima implementazione soprattutto grazie alla comoda (e veloce) rilevazione degli eventi.

Ma per questo settore è più che sufficiente una rapida scorsa ai programmi presenti nel package, tra i quali spicca uno **Sprite Editor** di tutto rispetto, realizzato, ovviamente, in Basic Amos.

Suono

La stessa versatilità, peraltro solo accennata, dei presìdi grafici, è riscontrabile a proposito del suono.

Amos, oltre alle consuete capacità di emettere **note** e **fonemi**, è in grado di eseguire completi brani musicali, purché in un suo formato (`.abk`), tramite il comando **Music**, e naturalmente in perfetto **multitasking**, mentre il programma sbriga altre faccende.

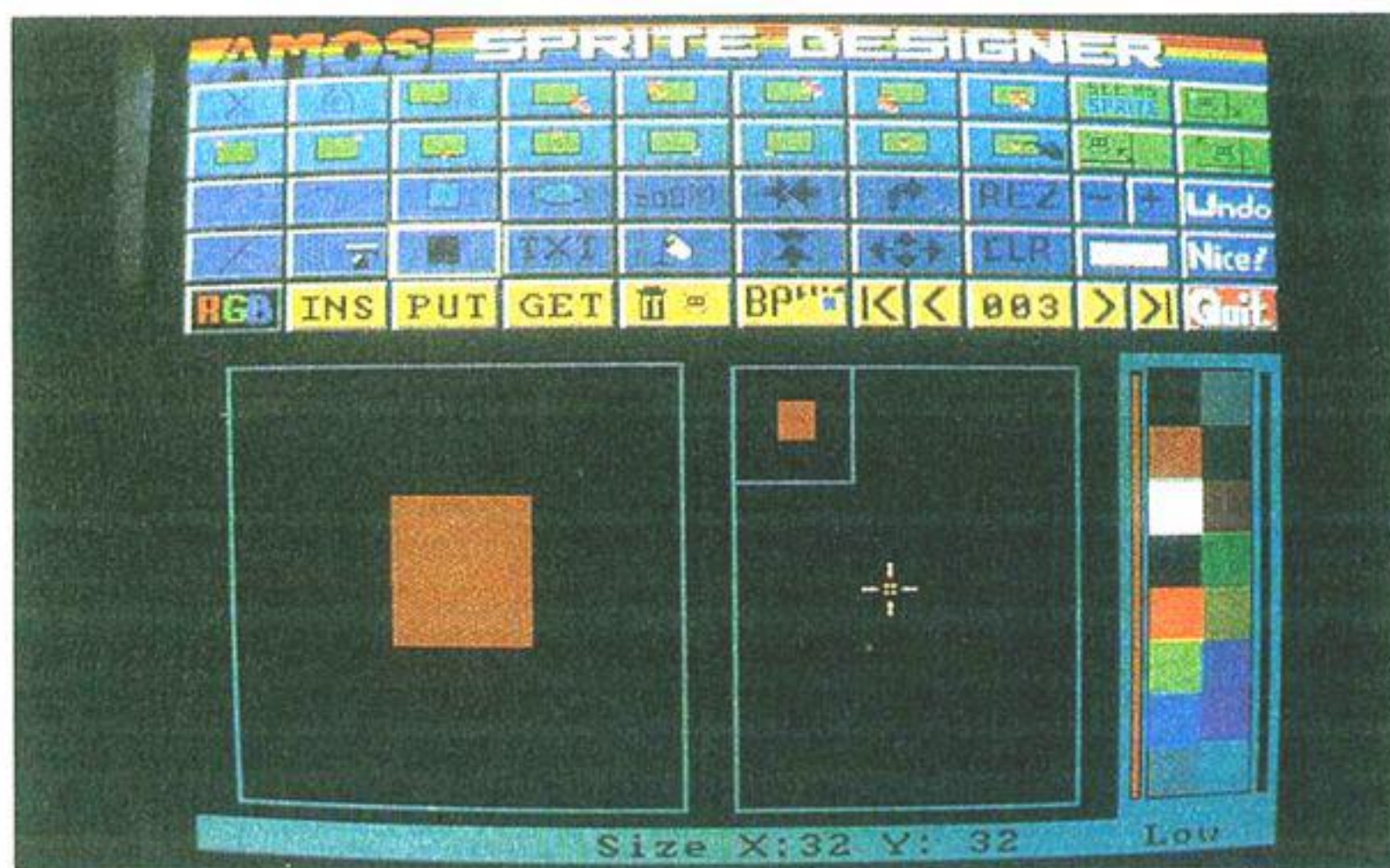
Se si considera che esistono in circolazione dei programmi (in Amos, guarda caso) che convertono i formati più svariati, si immagini cosa si può tirar fuori da un banale listatino.

Anche qui, inoltre, l'interprete è dotato di suoi effetti speciali: **Boom** (lo dice la parola stessa) per generare un suono simile ad un'esplosione, **Shoot** (colpo di pistola) e **Bell** (campana).

E, pur restando obbligatoriamente sul vago, si potrebbe continuare così per decine di pagine: eccellente gestione dei **menu**, facile intercettazione dei movimenti del **joystick**, e tante, tante altre delizie per chiunque non disdegni di accostarsi al Basic, ma anche per programmatori di linguaggi più complessi.

In conclusione non resta che accennare a due ultime "chance" di Amos. Una si chiama **Amal**, una specie di linguaggio interno ad Amos, che in pratica consente di compilare in memoria particolari sequenze di comandi dedicati all'animazione, ottenendo una velocità di esecuzione assolutamente paragonabile a quella dell'Assembler.

L'altra è la possibilità di rendere (quasi) autoeseguibili i programmi Amos, ovvero di affrancarli dalla presenza fisica del package. In concreto, si tratta di utilizzare **Ramos**, un programma del Pubblico Dominio incluso nel disco **Extras** di Amos 1.2, che contiene anche altre utility di tutto rispetto, come per esempio uno **Sprite Grabber** per creare uno sprite estraendone il disegno da una qualsiasi schermata Iff.

Il tutto, come già detto, in spasmodica attesa del promesso compilatore per un Basic... al di sopra del Basic.

di Domenico Pavone

Amiga Audiomaster III, per una musica "totale"

Per chi ama la musica, e ama Amiga, ecco un programma da non lasciarsi sfuggire

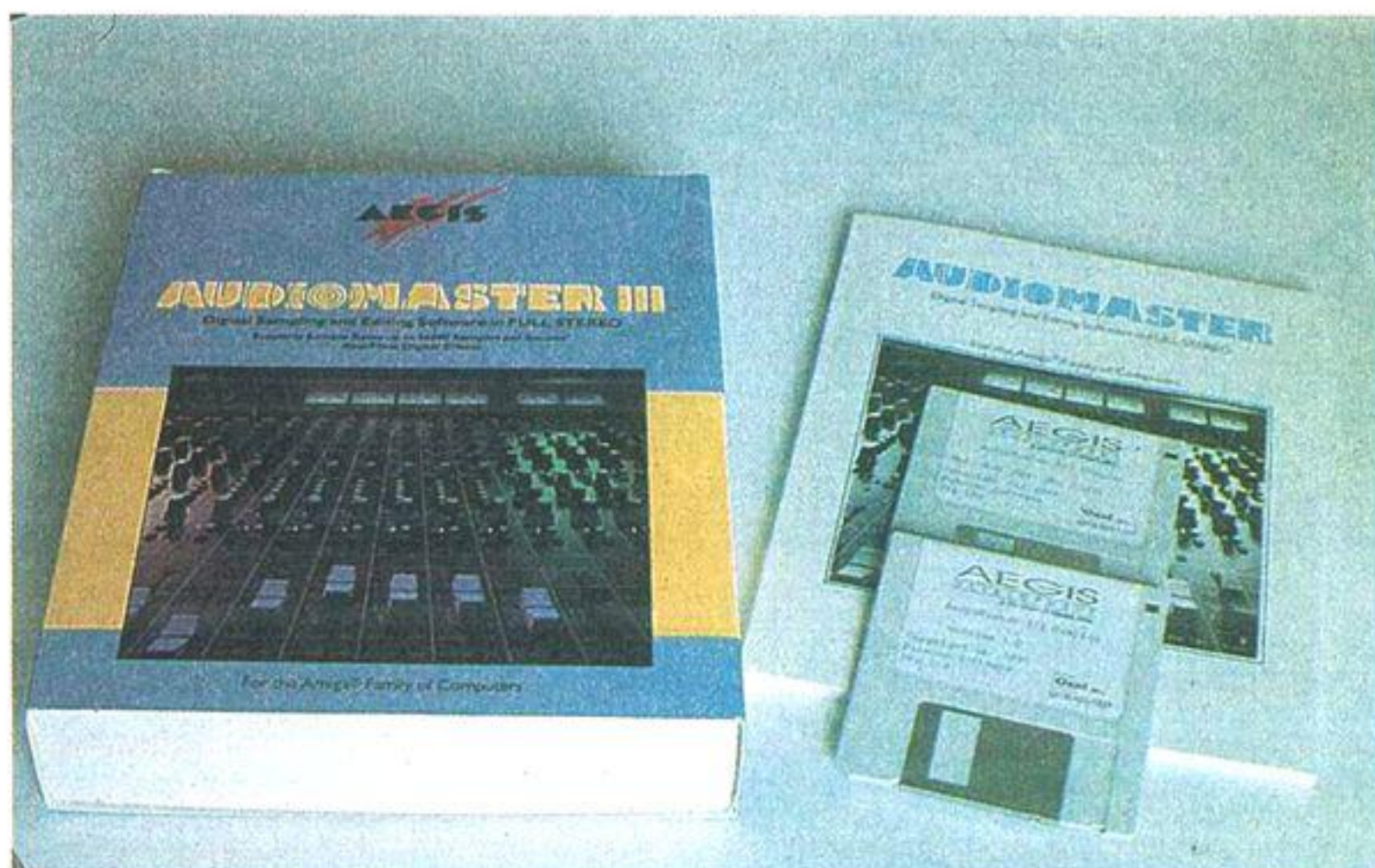
Dopo tanto parlare (o meglio, scrivere) di musica e suono in funzione di quelle meravigliose invenzioni che sono i digitalizzatori audio, non si poteva rompere completamente con quella che è diventata quasi una tradizione, lasciando a bocca asciutta i musicofili ad oltranza.

Pur non trattando, questa volta, di nuovi o semplicemente diversi accessori hardware, ma a ideale completamento delle prove effettuate nei numeri scorsi, ci occuperemo dunque di Audiomaster III, un software espressamente dedicato alla **digitalizzazione audio** (ma non solo), divenuto pressoché un *must* del settore grazie soprattutto alla sua vasta compatibilità con l'hardware in commercio, ivi compreso quello da noi esaminato nei numeri scorsi. Oltre che alle elevate prestazioni che lo caratterizzano, naturalmente.

Prodotto dalla nota **Aegis/Sonix**, Audiomaster è un pacchetto unicamente software dalle molteplici attitudini. E' certamente un editor del suono grazie alle potenti e decisamente semplici proprietà di manipolazione delle frequenze audio, ma è anche molto di più. Anzitutto perché comprende, oltre al programma principale, una serie di utility di tutto rispetto sulle quali ci soffermeremo tra breve. Inoltre, ed è forse questa la caratteristica più saliente, può interagire con una vasta gamma di **campionatori hardware** tanto direttamente che indirettamente: nel senso che può essere adoperato direttamente in congiunzione con i digitalizzatori, o accettare in input (con un banale Load) i files da essi prodotti.

Soprattutto nel primo caso, il vantaggio è evidente: di solito, l'hardware è corredato di software non molto sofisticato sotto l'aspetto dell'editing, in quanto viene prediletto il campionamento vero e proprio. In questi casi, Audiomaster può rappresentare un'alternativa molto valida. Tra i suoi numerosi menu accessibili dalla barra di schermo, è possibile infatti selezionare una scelta **Sampler**, dalla quale si accede ad un chiaro **requester** che consente di attivare "in proprio" la digitalizzazione. Chiaro che, in questo caso, deve già essere connesso all'Amiga l'hardware adatto, a sua volta debitamente connesso alla fonte sonora (*microfono, hi-fi, tape*, eccetera).

Notevole la compatibilità con i vari prodotti, e questo nonostante la stessa Aegis produca un suo digitalizzatore, il *Sound Master*, che speriamo di avere presto tra le mani. Si può quindi adoperare Audiomaster con i "classici" **Amas** e **Master Sound** della Microdeal (ci siamo occupati del primo pochi numeri fa), nonché **Perfect Sound, Future Sound, Soundscape** ed altri. L'input del "sam-

in vendita anche per corrispondenza presso FLOPPERIA Viale Monte Nero, 15 20135 MILANO tel. 02-55180484

pling" (la digitalizzazione) viene visualizzato in forma d'onda dall'opzione **Monitor** interna al requester Sampler, mentre due semplici cursori adattano Audiomaster alle caratteristiche dell'hardware.

Inoltre, è possibile decidere tramite l'opzione **Vox** on/off se far iniziare il campionamento in modo automatico alla ricezione in input del primo segnale audio, oppure affidarlo ad un click manuale del **mouse**. In qualunque momento, agendo sul pulsante **destro** dello stesso, si potrà inoltre interrompere momentaneamente la digitalizzazione. Va detto che tutte queste fasi, e soprattutto l'impostazione dei parametri per ottenere i migliori risultati, sono spiegate molto estesamente sul **manuale** a corredo del programma, alquanto chiaro anche se **in inglese**.

Nulla vieta, comunque, che si proceda alla digitalizzazione sfruttando il software a corredo dell'hardware, per poi caricare, tramite l'opzione **Load** di Audiomaster, un file **IFF**. Inutile dire che questo può essere **mono**fonico o **stereo**fonico, ma in ogni caso il programma è in grado di passare da una modalità all'altra convertendo di conseguenza il tipo di suono (!).

Come se non bastasse, Audiomaster può addirittura operare uno **scan** della memoria (submenu **Ram Scan** della voce Load), permettendo in tal modo di caricare anche un eventuale contenuto sonoro precedente al lancio del programma stesso, per poi salvarlo su periferica tramite l'opzione **Save Ranged Data** del menu **Project**. In questa fase l'editing in memoria è inibito per prevenire eventuali danneggiamenti al programma, ma come ovvio il file prima salvato può poi essere ricaricato e trattato come più aggrada. Per inciso, tutti i **Save** su disco possono essere effettuati, oltre che in modalità normale, in formato compresso, con notevole risparmio nell'occupazione del supporto megnetico.

Quanto finora detto, non può comunque che essere considerato una premessa: è infatti l'editing il settore in cui Audiomaster eccelle, soprattutto in considerazione dell'estrema facilità con cui è possibile ottenere risultati davvero notevoli.

L'Editor

La schermata di lavoro è occupata in massima parte da un **display oscilloscopico** nel quale vengono visualizzate, sotto forma di onda, le frequenza del suono o brano musicale caricato, mentre uno spartano ma completo pannello di controllo occupa la parte inferiore.

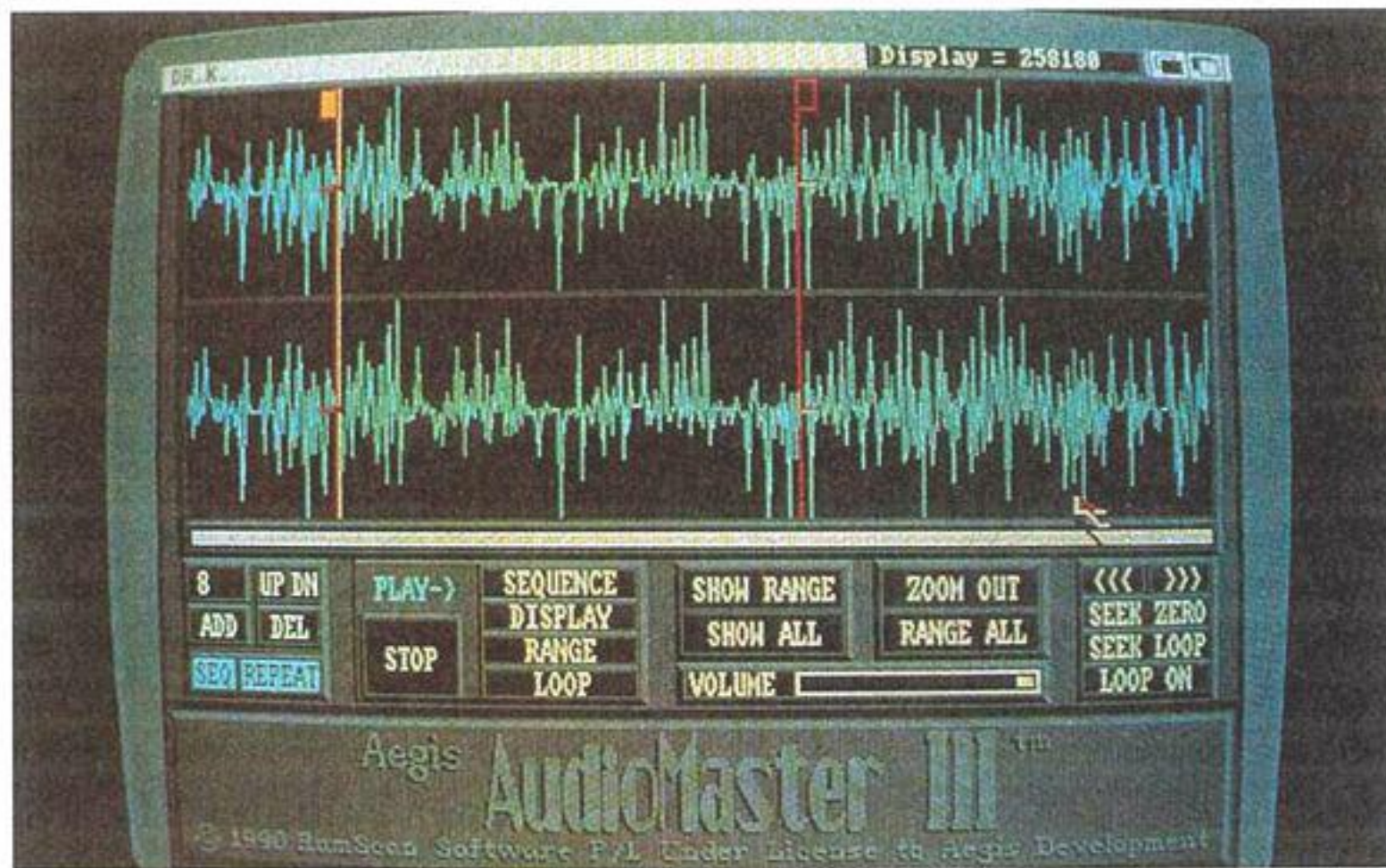
Nell'angolo in alto dello schermo, si ha inoltre una costante visualizzazione del numero di byte che corrispondono al sonoro visualizzato nell'oscilloscopio, ma la stessa indicazione può essere modificata da menu in modo da comprendere anche la durata in secondi delle sezioni d'onda selezionate. Dal pannello di controllo è possibile azionare uno **zoom** per cogliere dettagli sempre più particolari del suono, fino a giungere alle forme d'onda più essenziali, che diventano così più facilmente modificabili "a mano libera" (opzione **Edit Freehand** del menu **Edit1**) agendo con il mouse come se si stesse utilizzando un tool grafico.

Similmente a quanto avviene nei text editor, si può inoltre selezionare una porzione del display definita **Range** (oppure tutto con **Range All** dal pannello di controllo) che verrà mostrata in reverse, e sulla quale potranno essere applicate le più comuni (e note) funzioni di editing: **Cut**, **Copy** e **Paste**, che sfruttano un buffer di memoria ampiamente miscelabile con quanto presente nel display.

Sempre con pochi movimenti e click del mouse, è poi possibile accedere ad una delle più importanti *feature* di Audiomaster, l'inserimento di **loop** prestabiliti. Sul display sono infatti disponibili due barre verticali liberamente spostabili, all'interno delle quali può essere definito un loop con numero di iterazioni variabile. Per semplificare: si supponga di avere digitalizzato un breve brano musicale contenente la battuta ritmica di un "quattro quarti" di batteria. Ebbene, anche da un brano molto breve si può in pratica tirar fuori qualcosa di molto più completo ricorrendo al loop. Per esempio, si può racchiudere tra i delimitatori solamente la **battuta** ritmica sul piatto, ed espanderne la **durata** a proprio piacimento, personalizzando ed ampliando il brano originale. Caratteristica, questa, che può dimostrarsi ancora più efficace quando, piuttosto che un brano musicale, si intenda sfruttare un singolo campionamento sonoro, per esempio la nota di un pianoforte, che può essere dissezionata e modificata in ogni sua componente.

Tutte le modifiche possono essere ascoltate "in diretta" grazie ad un completo set di opzioni per il **Play**: si può riascoltare solo quanto mostrato nel display, o tutta la sequenza creata/digitalizzata, o ancora solo il Range o il Loop. Superfluo aggiungere che tutte le modifiche possono poi essere salvate in formato Iff, oppure in formato **Sonix compatibile**.

Resampling

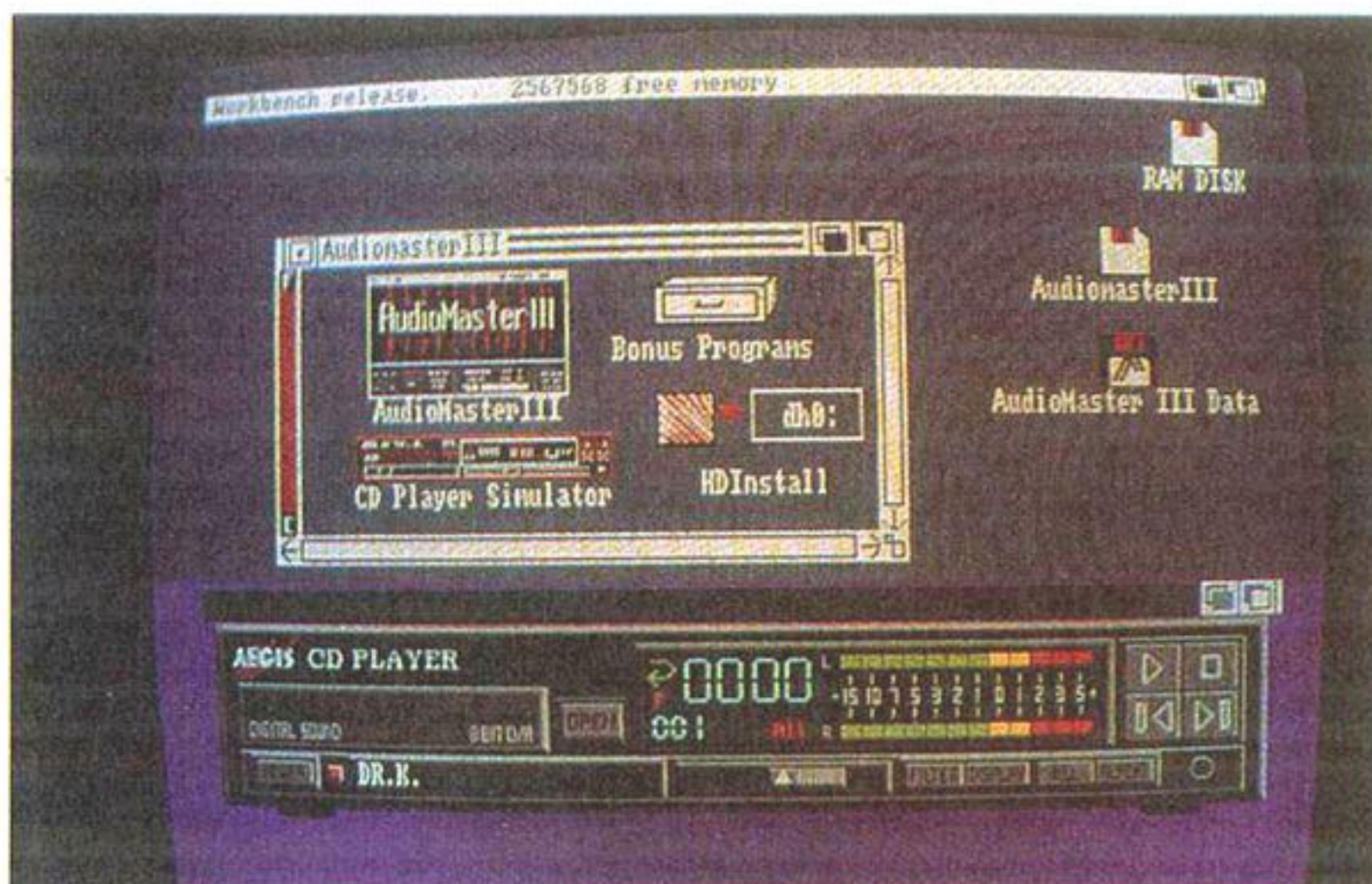
In un editor di questa stazza, non possono naturalmente mancare opzioni particolari, tra le quali spiccano l'inserimento di un **effetto eco** al sonoro (regolabile nelle sue componenti), la miscelazione tra suoni diversi, ed altre modifiche globali apportabili al brano digitalizzato, come per esempio l'ottava.

E fin qui, nonostante l'alto livello qualitativo, si può dire che molte delle opzioni di Audiomaster possono anche essere rintracciate altrove. Difficilmente tutte assieme, sarebbe necessario aggiungere, ma c'è una proprietà che rende questo programma del tutto unico: la possibilità di redigitalizzare il tutto in base a nuovi parametri.

Il che si può tradurre nel semplice **cambiare ottava** all'intero brano, ma soprattutto nella possibilità di ottenere sequenze audio di fedeltà impensabile in rapporto al cosiddetto **rate**. I suoni (o meglio i brani digitalizzati), infatti, quanto più lunghi sono (ovvero: quanto più lenta è la "registrazione") tanto più perdono di definizione. La causa di ciò è imputabile al rapporto tra **rate** e **pitch**.

Ebbene, entrambi i parametri sono rimodificabili adoperando Audiomaster (opzione **Tune Waveform** del menu **S/Fx**), per cui si può (sempre a titolo di esempio) abbassare il pitch di un campionamento, e far rieseguire la digitalizzazione dal software, ovviamente senza più bisogno dell'hardware.

In una sola parola: notevole.

Le Utility

Oltre al programma vero e proprio, non possono non essere citate altre utility che vengono fornite assieme al package.

Prima tra tutte il **Cd Player Simulator**, che consente l'ascolto autonomo di più brani musicali in sequenza rapida (memoria permettendo), agendo col mouse per azionare **filtro**, **interruttore**, **volume**, **display**, e tutte quelle regolazioni di norma presenti in un vero apparecchio di riproduzione.

Se il Cd Player può essere considerato un optional, non altrettanto si può dire degli altri programmi, di estrema utilità. Primo tra tutti **Playsound**, anch'esso un riproduttore multibrano, ma utilizzabile da ambiente Dos, e quindi particolarmente adatto per personalizzare propri dischetti ricorrendo a qualche banale batch file. Playsound può tra l'altro essere attivato con un'opzione **-w** che porta in secondo piano la sua finestra, evitando così di interferire con quanto presente sullo schermo: in altre parole, un sistema di visualizzazione grafica con sottofondo musicale (multiplo!) garantito.

A questo file si accompagnano poi **Makestereo**, in grado di fondere due files (IFF oppure **raw**) mono in uno unico, con ognuno dei due di origine che costituiranno un canale stereo; e ancora **Make Row**, più consueto per chi bazzica in ambiente Pubblico Dominio, utile per trasformare in dati finalizzati alla programmazione la sequenza digitalizzata.

E non è ancora finita. Con **Mergeiff** si possono attacchare due diversi brani memorizzati in IFF, molto utile se per esempio si dispone di poca memoria e non si possono trattare brani musicali particolarmente lunghi. In questo caso, è possibile operare due digitalizzazioni più brevi, e ricorrere poi a Mergeiff per avere un unico file su disco.

Infine **Oscilloscope**, un programma dotato di interfaccia grafica molto simile all'Audiomaster vero e proprio, ma limitato alla sezione display, che consente piccoli riaggiustamenti del sonoro da inviare all'output di Amiga.

Ed è tutto... ma non proprio tutto. Audiomaster può essere esplorato appieno solo dal vivo, e la carta stampata non può rendere piena giustizia ad un programma pressoché obbligato in ogni softeca Amiga che si rispetti.

di Gregor Samsa

Video, video, che passione!

Due soluzioni hardware per accostarsi facilmente alle più potenti risorse di Amiga

Le pagine di questa rubrica, di norma, sono dedicate alla descrizione di accessori delle più svariate categorie.

Con una costante presenza, negli ultimi tempi, di interfacce preposte al trattamento di segnali audio. Per pareggiare il conto, e soddisfare altri tipi di richieste pervenuteci, ci occuperemo stavolta di due strumenti riservati al trasferimento di immagini dal mondo esterno a quello gestito dal nostro amato computer, con due diverse modalità di approccio.

In altre parole, vedremo in azione un vero **digitalizzatore video**, in grado di trasformare l'input da una telecamera in schermate manipolabili da qualunque tool grafico, e un accessorio un po' particolare, capace di **miscelare** un segnate video composito esterno al normale schermo di Amiga, primo passo per chi volesse avventurarsi nel mondo del Desk Top Video amatoriale.

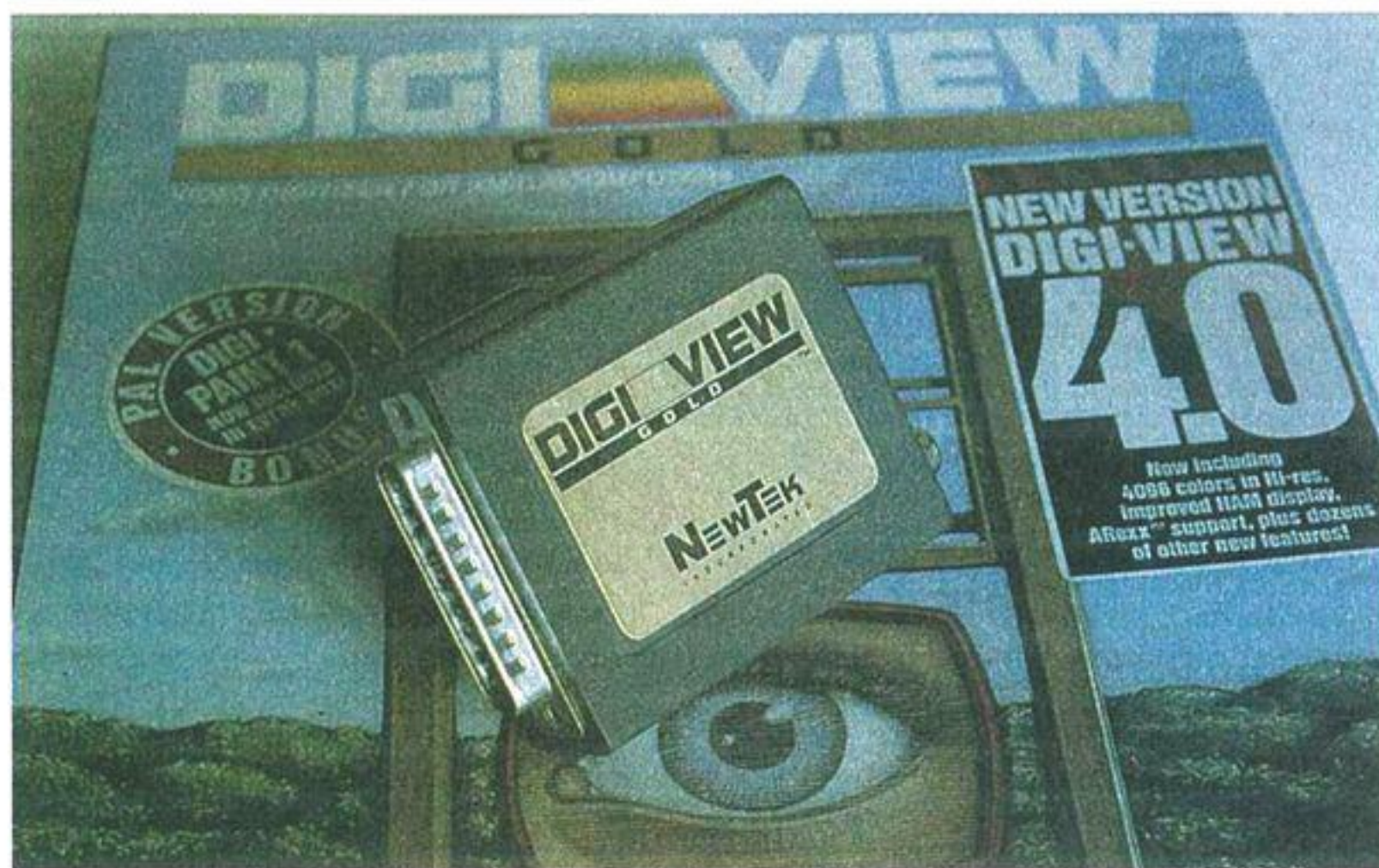
Come si può immaginare, ci si sta rivolgendo ad un settore dagli sviluppi infiniti, tra l'altro in costante espansione anche in ambiente professionale. Senza dovere a tutti i costi raggiungere questo livello, e soprattutto restando ancorati ad una fascia di acquisto accessibile ai comuni mortali, ecco dunque due possibili soluzioni per risultati tutt'altro che disprezzabili, pur se non paragonabili a quelli ottenibili... spendendo qualche milione in più.

Minigen

A prima vista, si rischia di scambiare questo accessorio per un comune modulatore, uno di quegli aggeggi che permettono di collegare Amiga ad un apparecchio TV piuttosto che ad un monitor. Immediata precisazione: simile nella forma e nelle dimensioni, ma non certo nelle prestazioni. A parte la *somiglianza* del contenitore, qualche altra differenza è comunque rilevabile anche ad un esame non approfondito. Intanto, sulla superficie superiore, si nota subito la presenza di un selettore a tre posizioni marcate **Mixed**, **Picture** e **Graphics**. Alle due estremità longitudinali si trovano invece gli ingressi hardware: un connettore a **23 poli** da inserire nella **porta video** di Amiga, e dall'altra parte **due** prese video **RCA** nelle quali innestare i cavi di input ed output del segnale video. Come dimostrato dall'esiguità del manuale in dotazione, Minigen è installabile con estrema facilità su Amiga, e può anche esservi lasciato in permanenza. I suoi circuiti sono infatti in grado di riconoscere se è presente un segnale video in ingresso: se dovesse mancare, il controllo viene ceduto in ogni caso al video di Amiga, che non subirebbe altre interferenze.

C'è però da dire che questa soluzione può rivelarsi utile solo per chi normalmente non adopera un monitor sfruttando l'output Rgb, quello che per esempio nei Commodore **1084** è riferibile alla presa **Scart**, molto più fedele nella riproduzione grafica. Il perché è presto spiegato, cominciando ad entrare nel merito del funzionamento dell'accessorio.

Il Minigen appartiene a quella categoria di hardware chiamata **Genlock**, ma gestisce solo segnali videocompositi, la cui sigla è **CVBS**. Esistono sul mercato dei genlock che sfruttano il segnale Rgb, ma la differenza di prestazione ne fa lievitare i costi oltremisura, rendendoli (al momento) accessibili ad una utenza prettamente professionale.

Minigen rappresenta comunque un ottimo compromesso tra qualità e prezzo, consentendo l'accesso al trattamento dei

video anche ad un hobbista evoluto. Ma cosa fa in pratica? Detto in poche parole: riceve in input un segnale video proveniente per esempio da un **videoregistratore**, e in uscita si potrà avere lo stesso segnale miscelato (e **sincronizzato**, sarebbe corretto dire) al video di Amiga. Il modo più semplice per vederlo in azione, consiste nel collegare l'output di Minigen direttamente al monitor di Amiga, sfruttando l'ingresso Cvbs in esso presente. Se si adopera un videotape, occorrerà comunque fornirsi di cavetti adatti, soprattutto se il registratore è fornito di sola presa Scart. Come ovvio, non può essere utilizzato il segnale proveniente dall'uscita modulata dello stesso, quella per intenderci che va a finire all'ingresso di antenna dei Tv. Questi cavetti, comunque, sono di facilissima reperibilità, in quanto commercializzati da tutte le rivendite di elettrodomestici.

Bene, supponendo di avere tutto il necessario, non resta che inserire Minigen nel connettore video di Amiga (a computer spento), infilare nella presa **Input** l'uscita video del tape, e collegare ancora la presa **Output** di Minigen all'ingresso Cvbs del monitor.

In questa configurazione, si può finalmente dare tensione ad Amiga, e apparirà il solito logo, anche se con *nitidezza inferiore* a quella cui si è abituati in Rgb. Chi adoperasse monitor Amiga (o similari), dovrà naturalmente agire sul pulsantino di selezione **Rgb / Cvbs** posto sul frontale dell'apparecchio, di solito siglato **Input I / Input II**.

Se a questo punto, con il videotape già in funzione, si porta il selettore di Minigen in posizione **Graphics**, in pratica... non succederà nulla di speciale. Lo schermo sarà sempre e solo impegnato dall'output video di Amiga. Se, invece, si seleziona **Picture**, sul monitor (o tv-monitor) risulterà visibile l'output del videotape, anche se questo non è impostato sul fermo immagine. Tra l'altro, c'è da dire, con una nitidezza notevole, soprattutto se la registrazione è in bianco e nero.

Chiaro che, ancora, non si è raggiunto il massimo delle aspirazioni. Non è certo necessario acquistare un Genlock per adoperare un videotape in modo pressoché normale. Il bello viene infatti se si porta la levetta di Minigen in posizione **Mixed**, e si ferma l'immagine del videotape in un punto voluto. Stavolta l'output del registratore costituirà lo sfondo dello schermo, mentre il primo piano sarà occupato dall'output di Amiga, sia esso il normale schermo Workbench come il più sofisticato programma di grafica, animazione, o videotitolazione.

La qualità del fermo-immagine sul videotape, come ovvio, riveste una certa importanza soprattutto nel trattamento di schermate a colori, ed è da evitare il più possibile la tipica oscillazione caratteristica di alcuni (vecchi) apparecchi. Da prove sperimentali, comunque, questa è più facilmente annullabile visualizzando l'output direttamente sul monitor, ovvero facendo passare il segnale prima attraverso Minigen, purché ovviamente il videotape consenta una regolazione del genere.

"Tutto" qua, che non è certo poco. Un esperto programmatore potrebbe anche gestire l'insieme grafico, in quando lo "sfondo", ovvero l'immagine importata dall'esterno, accuperà in pratica il **colore 0** di Amiga.

Ma il modo più semplice di sfruttare il tutto è ovviamente quello di manipolare a proprio piacimento l'immagine (sovrapponendovi titoli, animazioni, grafica, eccetera), e adoperare il connettore di output di Minigen per *trasferire il risultato verso un altro videotape*. Per chi non disponesse di due apparecchi di registrazione, l'input può ovviamente essere prelevato da una telecamera, e l'output inviato dove più aggrada: un videotape, ma anche apparecchiature più sofisticate.

I più evoluti potranno sentire la mancanza di feature come la dissolvenza o la possibilità di "grabbare" una singola immagine, ma... si provi a chiedere in giro quanto costa qualcosa come il **Video Toaster**.

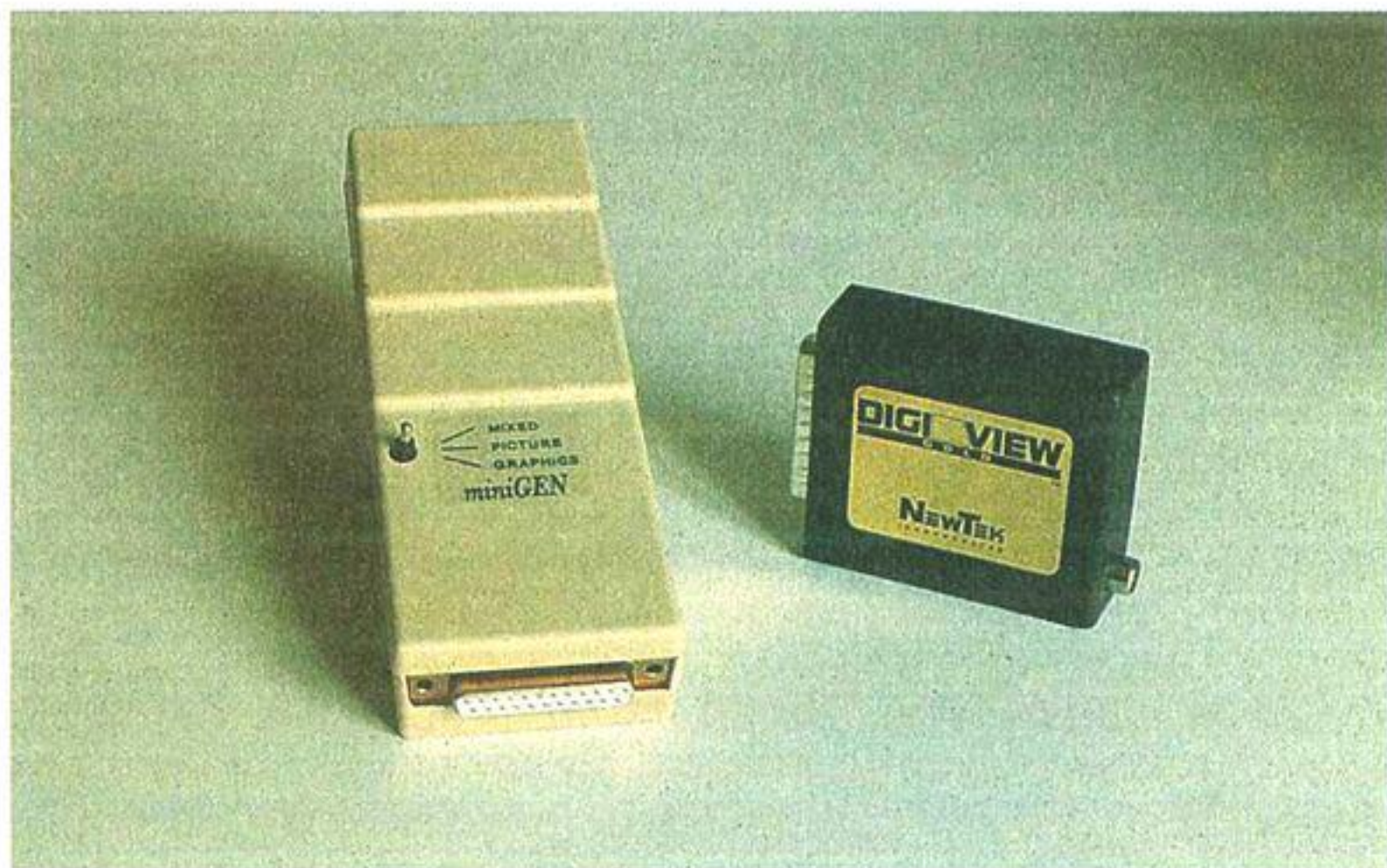
Per iniziare con un genlock, insomma, Minigen non è proprio male.

Digi View 4.0

Diversa è la filosofia di acquisizione (e utilizzo) di una immagine per questo prodotto della **Newtek**.

Digi View, infatti, è un *vero* digitalizzatore video, in grado di trasformale l'input di una telecamera in una schermata finalizzata all'uso in ambito Amiga. In altre parole, si possono creare videate personalizzate prendendo spunto da una fonte esterna, ma poi manipolandole come più si preferisce con tutti i tradizionali tool grafici che gestiscono il formato IFF, primo tra tutti **Digi Paint**, tra l'altro incluso al package di questa release **4.0** di Digi View.

Lo scopo finale può essere una semplice rappresentazione grafica, magari affidata ai vari **Slideshow** in circolazione, o l'inserimento di particolari sfondi o immagini in propri programmi.

Si è già detto che l'**input** va prelevato da una telecamera, ma c'è da aggiungere che l'immagine deve essere in ogni caso **fissa**. Il che significa che la stessa deve essere fissata ad uno stativo oppure ad un cavalletto, e puntata ad inquadrare una foto, un poster, un ambiente, o anche un oggetto. Il package, oltre ad

un'ampia manualistica, comprende l'accessorio di ricezione per Amiga, che va inserito nella **porta parallela**, ed è dotato di un connettore standard RCA al quale andrà il segnale **Cvbs** proveniente dalla telecamera. Come consigliato anche dai manuali, prima di effettuare questa connessione risulta utile testare l'immagine connettendo l'input direttamente al monitor (oppure a un Tv dotato di ingresso cvbs), in modo da ottimizzarne la regolazione.

Oltre agli elementi suddetti, ed ai floppy contenenti il software di gestione del Digi View e il Digipaint, nella confezione sono anche compresi due elementi essenziali ai fini della digitalizzazione a colori: una **staffa** da applicare alla telecamera, ed un supporto in acetato formato dai **tre filtri** fondamentali per la colorazione Rgb: **rosso**, **verde** e **blu**, più una sezione libera.

Questo supporto va connesso alla staffa in modo da poter ruotare davanti all'obiettivo, interponendo in pratica una filtratura tra la telecamera e l'oggetto da riprendere.

Il processo di digitalizzazione con Digi View richiede infatti una *scansione* delle tre componenti Rgb del colore, per giungere al risultato definitivo. Da un punto di vista pratico, la cosa è comunque molto semplice, e può anche essere ulteriormente semplificata acquistando il **Digidroid**, un motorino che fa ruotare i filtri senza alcun intervento manuale, pilotato da un'opzione presente nel software di gestione (menu **Digitize / auto**).

Una volta approntato l'apparato hardware, effettuate le connessioni, e "puntato" il soggetto da riprendere, non resta dunque che caricare in Amiga Digi View, settarne la risoluzione video non appena richiesto dal programma, e passare alla digitalizzazione vera e propria. Questa in pratica si risolve nell'attivare la telecamera, e selezionare uno dopo l'altro le opzioni Red, Green e Blue del menu **Digitize**, badando a far corrispondere il filtro di colore equivalente davanti alla telecamera. Quindi: prima di selezionare Red occorrerà ruotare il supporto davanti alla telecamera in modo da porre davanti all'obbiettivo il filtro rosso, quello blu prima di selezionare Blue, e quello verde prima di Green, giusto per chi non "masticasse" l'inglese.

Con il Digidroid, basterebbe invece scegliere **Auto** dallo stesso menu, e provvederebbe il programma (e l'hardware) a far ruotare il dispositivo quando necessario.

Il software di Digi View consente inoltre una vasta gamma di regolazioni (menu **Controls**), oltre ovviamente che il salvataggio dell'immagine digitalizzata in formato Iff oppure Rgb.

Quest'ultima opzione può risultare utile se si intende apportare altre modifiche con Digi View in un secondo tempo, in quanto i file prodotti risultano più ingombranti del familiare Iff. Particolarmente utile, inoltre, la voce **Histograms** del menu **Project**, che consente di visualizzare la percentuale delle diverse componenti Rgb (singolarmente), in modo da rendersi conto (tra l'altro) della luminosità dell'immagine.

Tornando ai controlli, questi possono riguardare tanto le caratteristiche del display (opzione **Control**), quanto le modalità di ricezione dalla telecamera (opzione **Camera**). Nel primo caso si ha a disposizione un pannello software ricco di scelte, tutte determinabili mediante l'uso del mouse. Prima tra tutte, il tipo di visualizzzione da associare all'immagine digitalizzata: **32 colori**, Halfbrite (**64 colori**), **Ham**, oppure in semplice **bianco e nero**. Si può anche optare per un'immagine in **negativo**, e regolarne il **dithering**, che corrisponde al processo di miscelazione di più colori per ottenerne uno nuovo. Non mancano poi i soliti cursori per determinare l'**intensità** delle tre componenti Rgb della **palette**, cui si accompagnano però altri in grado di variare l'intensità generale dell'**illuminazione**, del **contrasto**, e della **saturazione** cromatica della schermata.

Questo tipo di modifiche, dipendenti dal submenu Control, possono essere applicate all'immagine già digitalizzata, agendo sulla voce Display per vederne il risultato. Quelle invece che riguardano il menu Camera, vanno adoperate *prima* della fase di digitalizzazione, e consentono un **Fast Scan** per constatare rapidamente quale sarà l'effetto della digitalizzazione, che è comunque preferibile eseguire con attiva l'opzione **Normal Scan**. Per digitalizzazioni a colori, l'optimum lo si raggiunge comunque con **Slow / Color Camera**. Dallo stesso pannello di controllo, è inoltre possibile influenzare le dimensioni che assumerà l'immagine, o addirittura la sua posizione nello schermo, come pure ottimizzare il sincronismo video tra telecamera e monitor (**Tracking**), se necessario. Ad operazioni concluse, non resta che passare alla personalizzazione di quanto elaborato, sempre che lo si desideri. In questo caso, tra l'altro anche dall'interno di Digi View, è possibile attivare Digi Paint, uno dei più noti tool riservati alla grafica bitmap, come già detto incluso nel package. Questo programma è stato recensito sulla nostra rivista non molto tempo fa, per cui è inutile soffermarsi sulle sue caratteristiche, e comunque nulla vieta che, in alternativa, si adoperi il tool grafico cui si è più abituati (Deluxe Paint, Photon Paint, eccetera): la compatibilità non può che essere totale.

di Luigi Callegari

Diskmaster e gli altri disk editor per Amiga

Come ficcare il naso nei "solchi" dei nostri

Tutti sanno che l'uso di Amiga è incentrato sul cosiddetto ambiente **WIMP**, ovvero "**W**indow, **I**con, **M**ouse and **P**ointer" (finestre, icone, mouse e puntatore), che ne rende l'utilizzo particolarmente intuitivo (grazie a... Intuition!) e, generalmente, molto più rapido rispetto alla solita tastiera. Come prevedibile, dunque, i cosiddetti "editor di dischi", i **Pc Tools** per Amiga insomma, sono tutti basati su mouse, gadget e menu. In particolare, a differenza di quanto si vede negli analoghi programmi per Ms - Dos, i programmi per Amiga consentono di eseguire tutte le operazioni di manutenzione del sistema di archiviazione su disco senza (quasi) mai usare la tastiera. Operazioni come la formattazione dei dischi, la creazione e la ridenominazione di directory, lo spostamento e la cancellazione di file e directory od addirittura la visualizzazione di file IFF o la loro riproduzione sonora possono avvenire in modo molto più rapido ed efficiente rispetto alla digitazione da tastiera, specie se non si è formidabili dattilografi.

I programmi

Per Amiga esistono parecchi editor di dischi, alcuni di pubblico dominio (su **AmiGazzetta** ne verranno pubblicati alcuni) ed altri commerciali: **OpusDir**, **CLImate**, **DiskMaster**, eccetera. Lasciando la descrizione dei primi, spesso altrettanto validi di quelli commerciali, alla documentazione inerente ad AmigaGazzetta, abbiamo deciso di illustrare i concetti di utilizzo di base del più diffuso, e da noi preferito, programma commerciale, ovverossia di **DiskMaster V1.4** (scritto da Greg Cunnigham e prodotto dalla **Progressive Peripherals**).

Dal momento che i concetti di base del funzionamento di questo genere di programmi sono comuni per tutti, apprendendo il funzionamento di uno di essi ci si ritrova automaticamente, con poche differenze pratiche, ad intuire più facilmente l'utilizzo di *qualunque altro* programma analogo.

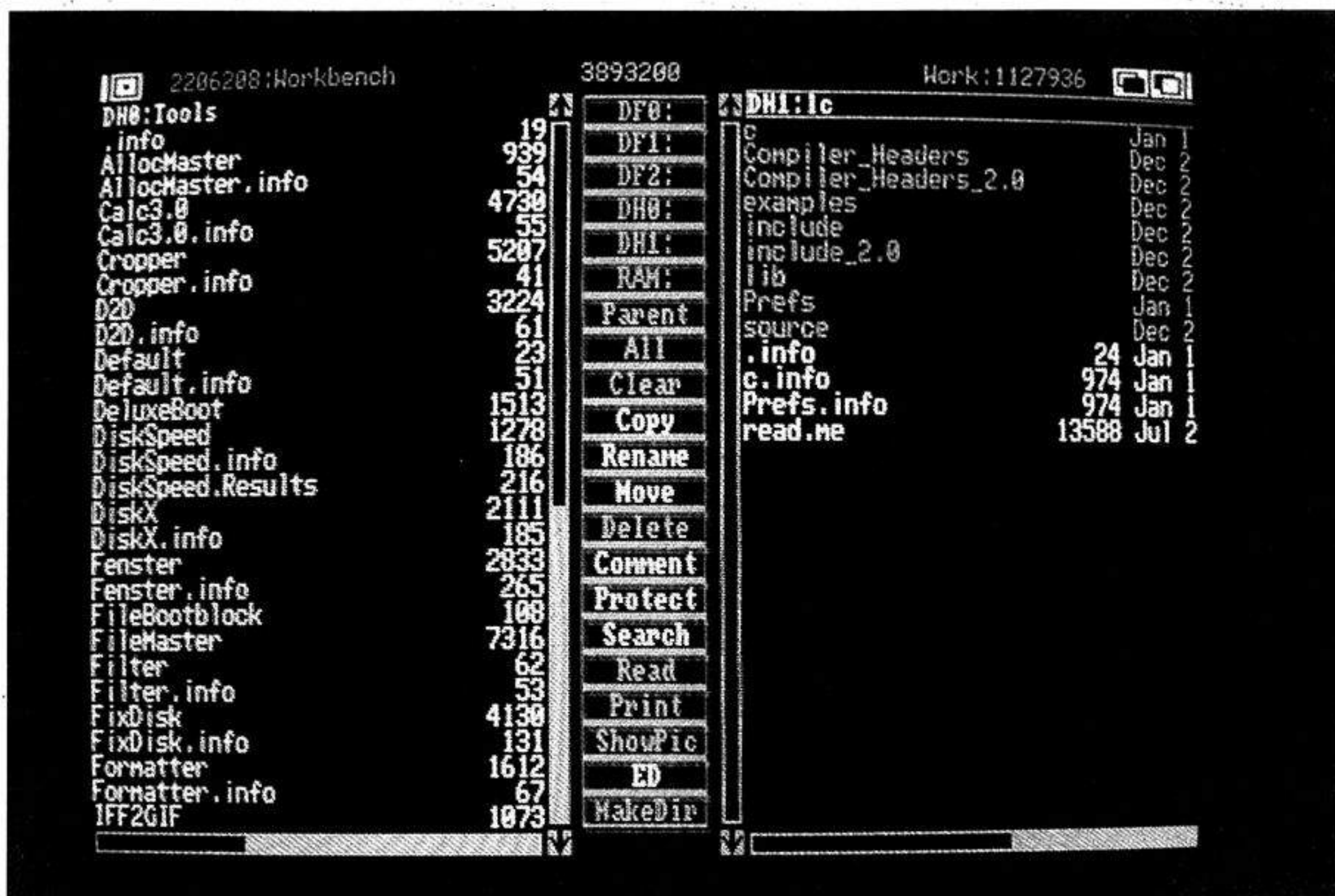
Ovviamente, è meglio avere già una certa conoscenza del modo di funzionamento di AmigaDOS: concetti come *directory ad albero*, *subdirectory* e *nome di volume* dovrebbero essere già note, ma cercheremo comunque di richiamarle in breve, a beneficio dei meno esperti.

Lo schermo di lavoro

Diskmaster è evidentemente studiato per essere sfruttato via mouse e gadget: al centro (verticalmente) troviamo infatti 21 *bottoni* che svolgono, ciascuno, almeno una funzione. Sono colorati in modo differente per consentire una migliore separazione agli occhi dell'utente. Non mancano tre menu con funzioni meno usate o di supporto, che descriveremo tra poco.

Diskmaster è un programma **personalizzabile**. La sua configurazione viene conservata nel file `Diskmaster.Config`, memorizzato nella directory associata al nome logico `Devs:`, ovverossia la directory "devs" del disco usato per avviare il computer (quello inserito alla richiesta della manina).

Dopo avere apportato le personalizzazioni volute, per registrarle nel file in modo che siano automaticamente riutilizzate dal programma ad ogni successivo avviamento, si deve scegliere l'opzione Configure/Save Config (ovvero, l'opzio-

ne "**Save Config**" del menu "**Configure**", il secondo di Diskmaster).

Configurazione

Per l'appunto, i primi sei gadget, partendo dall'alto, rappresentano i nomi dei volumi scelti dall'utente. Generalmente si inseriscono i nomi **DF0:**, **DF1:**, **DF2:** a seconda del numero di floppy disk drive posseduti, eventualmente **DH0:**, **DH1:** eccetera secondo il numero di partizioni del(l'eventuale) disco fisso installato, e **RAM:**, **RAD:** oppure **VD0:** per nomi dei dischi RAM installati.

Di default, Diskmaster scrive in questi gadget soltanto il **nome** dei floppy. Per aggiungere o modificare l'ordine delle assegnazioni si deve usare l'opzione **Configure/Set Device**. Così facendo nella barra del menu compare la richiesta *"Set device to Rename"*, che spiega come sia necessario cliccare su di uno dei primi sei gadget per consentire la ridenominazione.

Cliccato sul gadget interessato, compare un requester con un gadget di stringa che consente di scrivere il nuovo nome del device (RAD:, DF2:, DH0:, VD0: eccetera), da confermare cliccando su Okay (o premendo Return), operazione annullabile con Cancel od Abort.

Bisogna ricordarsi che i nomi di device di Amiga finiscono sempre col doppio punto (:) e sono lunghi soltanto **tre** caratteri.

Nel caso non bastassero i sei gadget, basta posizionarsi su di uno di essi e cliccare col pulsante **destro** del mouse: appariranno **altri** sei gadget "alternativi" a rotazione per ogni pressione del pulsante destro. Anche questi sono configurabili (e salvabili in Diskmaster.config) nel modo appena detto.

Il menu Configure

Altre opzioni configurabili riguardano le funzioni del secondo menu, Configure appunto. Con **Set Colors** appare un requester con otto colori, regolabili tramite appositi *slider* orizzontali. Si possono stabilire a piacere tutti i colori usati da Diskmaster: basta cliccare sul colore da modificare (riportati in rettangolini verticali con a fianco il valore esadecimale interno di Amiga) e poi agire sugli slider.

Per annullare le modifiche bisogna cliccare sulla scritta **Cancel**, per terminare le modifiche conservandole bisogna invece cliccare sul consueto gadget di chiusura della finestrella.

L'opzione Configure / **Resolution** consente di scegliere la risoluzione dello schermo usato da Diskmaster: bassa, alta o a "mezza altezza" (con visione del Workbench sottostante l'esecuzione del programma).

L'opzione Configure / **Small Font** usa un set di caratteri più piccolo, che consente di visualizzare più caratteri su di una stessa linea. Quando lo si attiva, compare un simbolo di *"spunta"* col sim-

I PICCOLI COMMODORE SONO CRESCIUTI

bolo Amiga, che rimane attivo sinchè non si riseleziona la stessa opzione.

L'opzione Configure / **Info Copy** è un altro commutatore che impone a DiskMaster di eseguire copie integrali di tutti gli attributi dei file e directory copiate. Normalmente, infatti, un file copiato assumerebbe lo stato dei flag di protezione (WERDSAP) di default (scelti tramite l'opzione Configure / Set Protect) e la data attuale, esattamente come avviene se non si usa l'opzione Clone nel comando Copy di AmigaDos.

L'opzione Configure / **Confirm** impone al programma di richiedere una conferma tramite requester prima di eseguire ogni operazione che comporti la cancellazione di un file o di una directory.

L'opzione **Set Pattern** consente di scegliere la maschera di selezione dei file nelle finestre. Ad esempio, per evidenziare tutti i file dotati del suffisso "**.doc**" si specificherà, nel requester che appare, "***.doc**".

La successiva opzione consente di stabilire il nome del comando "personalizzabile" riportato per penultimo (prima di Makedir) nella lista dei gadget. Noi, ad esempio, abbiamo inserito il nome **ED**, che consente di caricare automaticamente l'editor di sistema per redigere un file ASCII selezionato dall'interno di Diskmaster.

Lavorare con le directory

Diskmaster ha **due finestre verticali** ai lati della lista dei gadget di funzione. Queste sono destinate a contenere le liste dei file e delle directory contenute nel "**path**" specificato nell'apposito gadget stringa superiormente a ciascuna di esse, delimitato da una piccola cornice. Per indicare il path di una finestra vi sono vari modi. Ad esempio, possiamo cliccare con il mouse all'interno del gadget di stringa superiore ad una delle due finestre e poi scrivere, da tastiera, il nome del path, come ad esempio:

```
DF0:Avv
```

...che, premendo Return, impone a Diskmaster di analizzare la directory chiamata **Avv** assunta nel disco del drive interno e di presentarla sul video. La lista contiene prima tutti i nomi di directory in ordine alfabetico, poi in colore diverso (secondo la vostra configurazione cromatica) tutti i nomi di file in ordine alfabetico. Accanto a ciascuna directory vi è la data di creazione o dell'ultima modifica interna, seguita dai flag di protezione. Per visualizzare tutti questi dati si può agire col mouse sullo slider di scorrimento **orizzontale** collocato sotto ciascuna delle due finestre di lavoro.

Analogamente, agendo sullo slider **verticale** interposto tra ciascuna finestra e la lista dei gadget, si possono visualizzare i file o le directory che non compaiono sul video a causa del limitato spazio disponibile sulla finestra. I file sono accompagnati anche dalla loro lunghezza.

Tra l'altro, per completare il quadro delle informazioni numeriche, bisogna notare che, nella barra del menu sopra ciascuna finestra, viene riportato lo spazio libero (in bytes) nel volume selezionato e centralmente, proprio sopra i gadget di funzione, la quantità di Ram di sistema libera. Quando si selezionano dei file col mouse, l'indicatore numerico riporta non più lo spazio libero sul device, ma la lunghezza complessiva dei file selezionati.

Un altro modo per scegliere un path consiste nell'usare il mouse: per indicare il device basta cliccare all'interno di una delle due finestre di lavoro (la sinistra o la destra), che diventa così "corrente" o "selezionata" per la successiva operazione (il riquadro sul gadget di stringa che contiene il path si colora in modo diverso), poi si clicca su di uno dei gadget di volume configurati (DF0:, RAM: ...). Comparsa la lista dei file, per "scendere" in *una* directory basta cliccare **due volte** sul nome di *quella* directory col mouse: si vedrà nel gadget stringa aggiornare il nome del path e la finestra che presenta i contenuti della nuova sub-directory. Cliccando sul gadget **Parent**, sotto i gadget dei device, si risale di un livello nella gerarchia della directory.

Le funzioni che lavorano su di una sola directory usano sempre quella corrente. Ad esempio, per **creare una nuova directory** sul disco inserito nel drive **DF0**, bisogna cliccare su di una delle due finestre (quella con contenuti che non interessano), poi sul bottone con DF0 e infine su quello con **Makedir**. Nel requester che appare si scrive il nome della directory da creare e, battendo su Return o cliccando su **Okay** si ottiene la creazione della directory in quel volume. Allo stesso modo, ovviamente, si può creare una subdirectory: basta che nel gadget stringa sopra una delle finestre sia presente il nome di path completo interessato e che quella finestra sia stata resa corrente.

Operazioni varie

Per copiare un file, si deve tenere conto che il trasferimento avviene dalla finestra *selezionata* a quella *deselezionata*.

Ad esempio, se nella finestra di sinistra abbiamo il path **DF0:Archivio** che contiene un file "**uno**" da copiare in Ram Disk, dovremo prima di tutto portare la finestra di destra sul device **Ram:** nel modo prima visto, poi cliccare sul nome del file "**uno**" della finestra di sinistra e, per eseguire la copia, sul gadget **Copy**. Vedremo che dopo qualche istante la finestra di destra (deselezionata) si svuota e si ripresenta con il file Uno nell'elenco.

Ovviamente, si possono copiare in un colpo solo più file e directory: basta che prima siano selezionate adeguatamente in una delle due finestre prima di cliccare sul bottone "Copy".

La funzione **Move** esegue un'operazione analoga, ma cancella il/i file/directory dal device originale.

La funzione **Delete** cancella tutti i file selezionati. Si noti che chiede sempre conferma prima di cancellare directory che contengono dei file.

Per selezionare in un colpo solo tutti gli elementi riportati in una finestra, bisogna cliccare sul gadget **All**, mentre per deselezionarli bisogna cliccare su **Clear**. Se si è scelta una maschera di selezione tramite Configure / **Set Pattern**, All e Clear la useranno per determinare quali file scegliere.

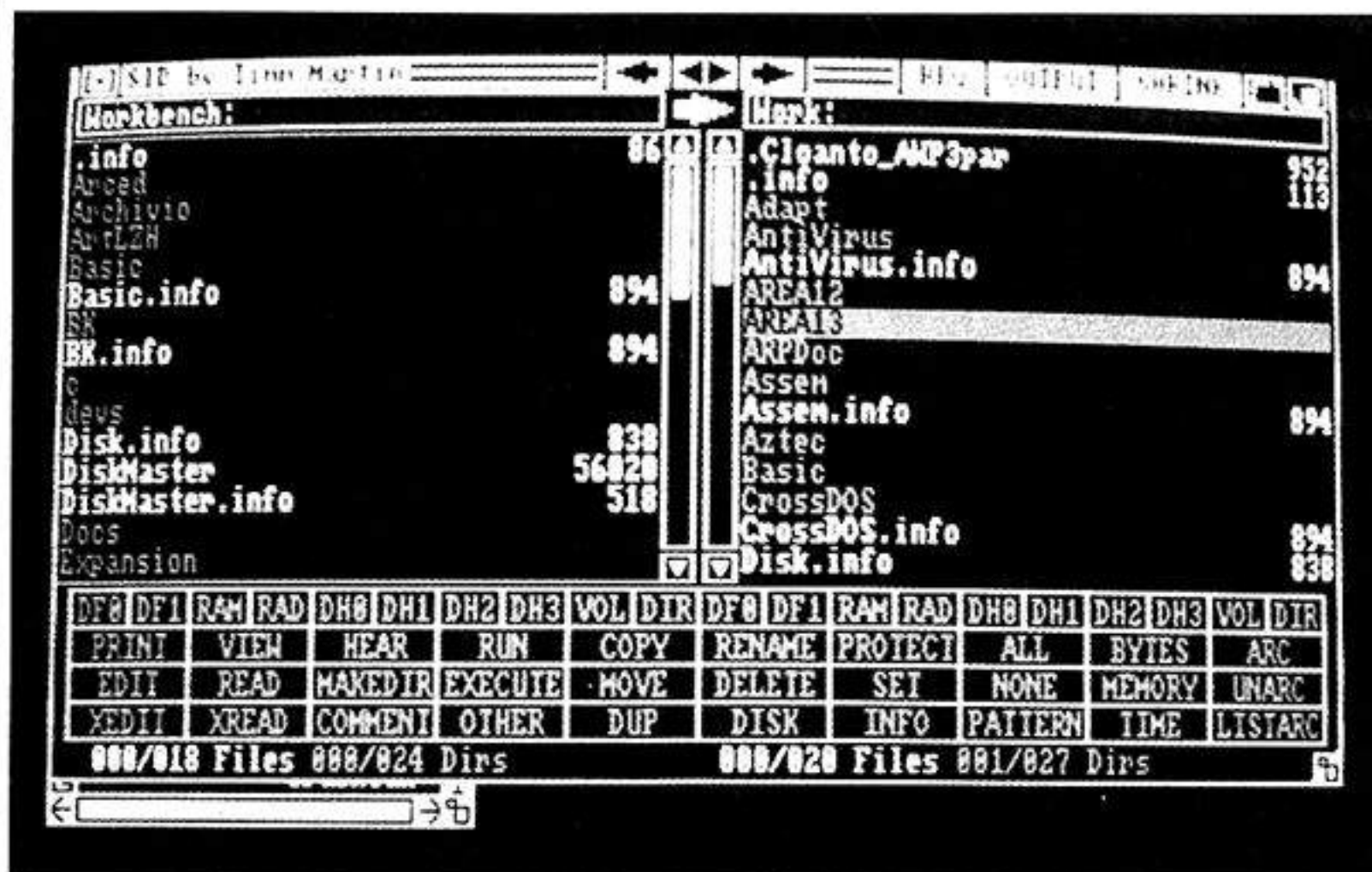
La funzione **Rename** consente di riscrivere il nome di tutti i file evidenziati tramite un apposito requester con gadget stringa, mentre **Comment** permette di specificare un commento (come si fa col comando *Filenote* del **CLI**) per i file e le directory specificate.

Il comando **Protect** consente di controllare singolarmente i flag di protezione di scrittura (**W**), lettura (**R**), esecuzione (**E**), cancellazione (**D**), archiviazione (**A**), Pure (**P**) e Script (**S**) tramite gadget, selezionabili col mouse, che riportano le iniziali.

La funzione **Search** ricerca, nella directory evidenziata, i file che hanno nel nominativo una certa maschera specificata con i soliti caratteri jolly "*" e "?". Dopo avere localizzato ogni occorrenza, viene richiesto se proseguire od annullare la ricerca.

Il gadget **ShowPic** visualizza il file IFF evidenziato in qualunque formato o risoluzione sia (per uscire bisogna cliccare nuovamente). Se si scelgono più file, verranno visualizzati in sequenza a seconda della pressione del tasto sinistro del mouse o della barra spaziatrice, indifferentemente. Cliccando col pulsante destro del mouse sul gadget ShowPic, questo si trasforma in **PlaySnd** e consente di generare in audio il suono rappresentato da un file in formato IFF sonoro.

Analogamente, il gadget **Read** si trasforma in **HexRead** se si clicca su di esso col pulsante **destro.** In ogni caso, visualizza in ASCII od in ASCII ed esadecimale, rispettivamente, il contenuto del file selezionato. Per scorrere durante la lettura basta spostare il puntatore del mouse verso l'alto o verso il basso (la velocità di scorrimento è proporzionale alla posizione del mouse rispetto al bordo inferiore o superiore della finestra).

Il gadget **Print** consente di stampare in modo piuttosto evoluto un file. Cliccando su di esso compare un requester che chiede quali **parametri di stampa** utilizzare. Ad esempio, si può richiedere di stampare anche il nome del file, l'ora e la data registrate come ultimo aggiornamento del file, il numero consecutivo di pagina, la qualità di stampa (Draft o NLQ) ed il numero di linee per pagina (variabile

cliccando sui gadget di maggiore > o minore < ai bordi). Per avviare la stampa si deve cliccare su Print, mentre, per annullare l'operazione, su Cancel. Si possono selezionare più file prima di agire sul gadget Print dallo schermo principale, nel qual caso i file verranno stampati consecutivamente, previa conferma tramite click sul gadget Print del requester per ogni File.

Menu Project

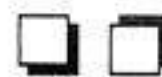
Sotto il menu Project sono raggruppate alcune funzioni supplementari di Diskmaster, di uso meno frequente rispetto a quelle raggruppate nei gadget sinora descritte.

La funzione **DiskCopy** esegue, come prevedibile, la copia di due dischetti. Dopo averla scelta dal menu a sipario **Project**, compare un minuscolo requester che richiede i drive sorgente (**From**) e destinazione (**To**). Per cambiarli basta cliccare sul nome del drive, ottenendo così la rotazione del numero (DF0, DF1, DF2, DF3). Per avviare la duplicazione si sceglie Okay, mentre per annullarla si deve scegliere col mouse Cancel.

La funzione **Format** formatta un dischetto. Compare un requester che richiede il nome da assegnare al disco (**Empty** per default), se occorre installarlo (**Install**) ovvero renderlo di tipo "bootable" e se occorre verificare la formattazione (**Verify**).

Per avviare la formattazione si deve cliccare su Okay (o premere Return), per annullarla basta scegliere Abort o Cancel.

La funzione **Print Dir** stampa tutti i contenuti di una directory correntemente selezionata.

La funzione **Workbench** apre e chiude il Workbench, per consentire il rientro ad esso o per risparmiare memoria, rispettivamente.

Il menu Archive

Il menu Archive comprende alcune funzioni legate all'uso dei programmi di archiviazione e compressione più diffusi per Amiga. Per usarle bisogna però avere inserito nella directory C del disco di sistema i programmi di pubblico dominio Arc, LHarc e Zoo (forniti su AmiGazzetta).

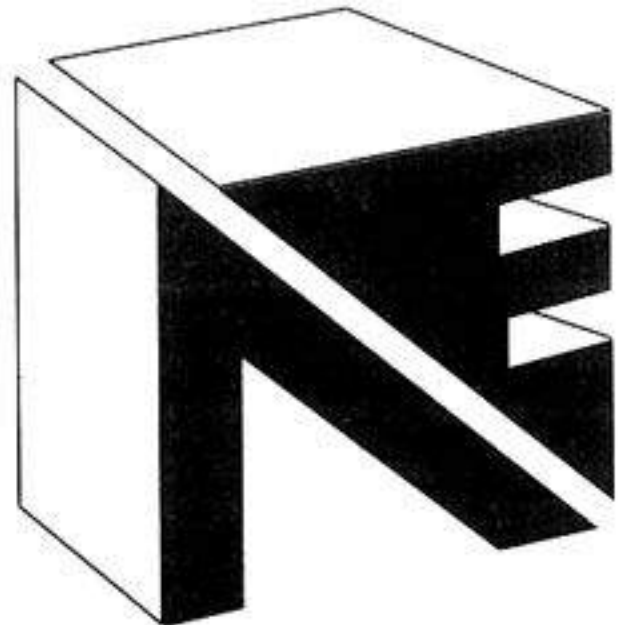
In questo modo è possibile estrarre, aggiungere o listare i file contenuti in archivi prelevando gli elementi dalle directory riportate nelle finestre, secondo i normali canoni di funzionamento di questi comandi, generalmente accessibili solo dal CLI.

NEWEL® srl

computers ed accessori
20155 MILANO via Mac Mahon, 75

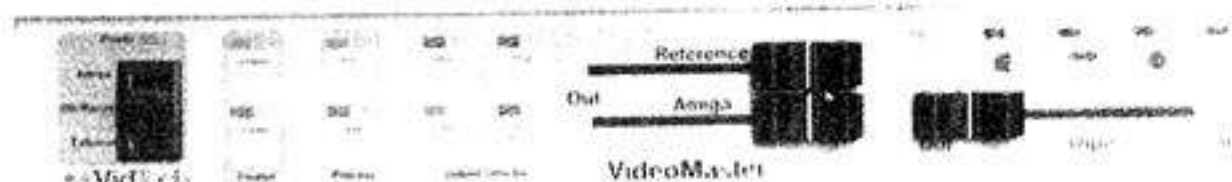
NEGOZIO tel.02/3 2 3 4 9 2
UFFICI tel.02/3 2 7 0 2 2 6
FAX 24h tel.02/3 3 0 0 0 0 3 5
UFFICIO SPEDIZIONI tel.02/3 3 0 0 0 0 3 6

ORDINA SUBITO! →

UNICA SEDE IN ITALIA

richiedi il nostro nuovo catalogo gratuitamente specificando il computer posseduto!

XC'E BULK E BULK NOI TI OFFRIAMO DISCHETTI DI QUALITÀ 3 1/2 DS-DD CERTIFICATI UNO AD UNO.

SONY CONF. DA 50 PZ. L. 900 CAD.
SONY CONF. DA 100 PZ. L.800 CAD.
SONY CONF. DA 200 PZ. L;700 CAD.
TUTTI I PREZZI IVA COMPRESA

VENDITA PER CORRISPONDENZA IN TUTTA ITALIA
EVASIONI ORDINI NELLE 24 ORE SUCCESSIVE ALL'ORDINE

VIDEOMASTER

Il più straordinario Genlock professionale per tutti gli amiga, Qualità Broadcast, S-VHS (luminanza cominanza separati) già in console effetti video.

ACCESSORI VARI

RAM ZIP 414256 (256 x 4) 80 ns. Per Amiga 3000 (8 per 1 MB) **L.18.000** cad
RAM ZIP 414400 (1 MB x 4) 80 ns. Per Amiga 3000 ([illegible] per 1 MB) **L.80.000** cad
Kit 2 MB per Hardisk A590 (espande la memoria ad 2 MB) **L. 189.000**
Kit [illegible] MB per HARDCARD 2091 Commodore **L. 199.000**
Kit 2 MB per HARDCARD GVP semplicissima installazione **L. 275.000**

- NOVITA' MONDIALE - NOVITA' MONDIALE -

Probabilmente il più piccolo HARDISK da 20 MB al mondo!!! - Si inserisce all'interno del AMIGA 500 - Alta velocità 20 MB drive - Formattato per amiga DOS - Autoboot sotto Kickstart 1.3 Autoparking - Compatibile con Adspeed accelerator - Semplice installazione

TUTTO QUESTO AD UN PREZZO ANCORA PIU' PICCOLO!!! MENO DI UN MILIONE

SUPER SYNCRO EXPRESS V. 3.0

Nuova versione del più potente copiatore Hardware, con nuovo CHIP castum, vi permette di effettuare copie di sicurezza ad uso strettamente personale della maggior parte del software protetto, opzione Quickcopy in meno di 1 minuto, semplice installazione, funziona su Amiga 500 & 1000.**L. 89.000**

• ESPANSIONI DI MEMORIA PER AMIGA 500, 1000 E 2000

Costruite con i migliori materiali, le nuove espansioni di memoria dell'ultima generazione usano i nuovissimi chip da 1 mbit che sono notevolmente più veloci, autoconfiguranti, slim line, e con 1 anno di garanzia!

- 512k per amiga 500 L. 85.000
- 512k per amiga 500 + clock L. 110.000
- 1,5 mb per amiga 500 + clock L. 240.000
- 4 mb per amiga 500 +clock L. 590.000
- 2 mb per amiga 1000 L. 450.000
- 2 mb espandibile a 8 per amiga 2000 + clock L. 390.000

UTILITY DISK DEL VALORE DI L. 50.000 IN OMAGGIO

NUOVE SCHEDE ACCELERATRICI "GVP A-3001" ANCORA PIU' VELOCI!!!!

Nuovo ora 33 o 50 Mhz. Accelera il tuo Amiga A2000 più di dieci volte, rendendolo notevolmente più veloce del nuovo A3000, corredato di 4 MB di ram a 32 Bit, e coprocessore 68882, completo di controller per Hard-disk.

NELLA VERSIONE 33MHZ. L.3.390.000
NELLA VERSIONE 50MHZ. L.4.990.000
KIT. SUPPLEMENTARE PER ALTRI 4 MB A 32 BIT. L. 899.000

PAL GENLOCK 3.0

Nuovo genlock semiprofessionale, per tutti gli amiga, con nuove funzioni. Un nuovo circuito facile e gradevole da usare consente una perfetta sovrapposizione OVERLAY computer/video, il segnale video trasparente del segnale del Computer. Nella condizione SUPER IMPOSE si ottiene la sovrapposizione completa del segnale computer, compreso lo sfondo, con apparente funzione di mixer video. Un sofisticato circuito elimina l'effetto arcobaleno dal colore bianco. Una serie accurata di tarature rende molto stabile il segnale congiunto GENLOCK/VIDEO. Ultimo e di grande importanza un nuovo circuito di uscita di "potenza" permette l'adozione standard di un cavo di collegamento di almeno 50 cm. Queste sono solo le novità oltre la dotazione base della precedente versione, come il fader (dissolvenza). Il genlock è particolarmente indicato per la titolazione di videocassette.
L. 390.000

MODEM RS-232 STANDARD HAYES COMPATIBILI:

ESTERNO SMART 300/1200 (V21-V22) L. 179.000
ESTERNO SMART 300/1200/VIDETEL (V21-V22-V23) L. 249.000
ESTERNO SMART 300/1200/2400 (V21-V22-V22 BIS) L. 249.000
ESTERNO SMART 300/1200/2400 con correzione errore MNP-5 L. 369.000
ESTERNO SMART 300/1200/2400/VIDEOTEL (V21-V22-V23-V22BIS) L. 399.000-
DISPONIBILI ANCHE TUTTE LE VERSIONI IN SCHEDA INTERNA!!!

ORA ANCHE PER A2000

AMIGA PC AT EMULATOR

ATONCEPC 286
EMULATOR AT
AMIGA 500 MULTITASKING

Lire 390.000

MANUALE IN ITALIANO

GARANZIA 12 MESI

- **DRIVE INTERNO PER AMIGA 500 L. 149.000**
- **DRIVE ESTERNO PER AMIGA 500/1000 PASS. + DISCONNECT L. 149.000**
- **DRIVE INTERNO PER AMIGA 2000 (COMPLETO) L. 139.000**
- **DRIVE ESTERNO PER AMIGA 500/1000/2000 DA 5"1/4 L. 249.000**

UTILITY DISK DEL VALORE DI L. 50.000 IN OMAGGIO

ORA DISPONIBILE ANCHE SEGA MEGADRIVE CON MOLTI GIOCHI

tutti i nostri prodotti sono coperti da garanzia di 12 mesi

TUTTI I PREZZI SONO IVA 19% COMPRESA

by Mirko Lalli

Una clessidra per la tua Amiga

Ricordate la sfida lanciata, da queste stesse pagine, nel lontano n. 79? Bene un nostro lettore...

Se volete sapere per quanto tempo avete utilizzato il vostro Word Processor preferito; se dovete conoscere il tempo che vi ruba il programma di Raytracing; o semplicemente desiderate vantarvi con gli amici del tempo trascorso a pasticciare con la grafica, allora continuate a leggere questo articolo, che rappresenta una risposta alla sfida lanciata nel numero 79.

La sfida

Sul numero 79 di questa rivista è stata lanciata una sfida riguardo la realizzazione di una procedura che riuscisse a **memorizzare, in un apposito file, l'ora in cui cominciamo a utilizzare un qualsiasi programma e l'ora nella quale lo stesso programma viene abbandonato.** Il tutto per avere una stima del tempo trascorso nell'uso di quel particolare programma. In queste pagine compare l'intera procedura con relativo programma, in Amigabasic, che è il decodificatore del file creato dalla procedura Batch.

Il file Batch (o script) è tanto semplice quanto funzionale, dal momento che aggiorna accuratamente il file di dati in modo totalmente automatico e trasparente per l'utilizzatore.

Il file batch

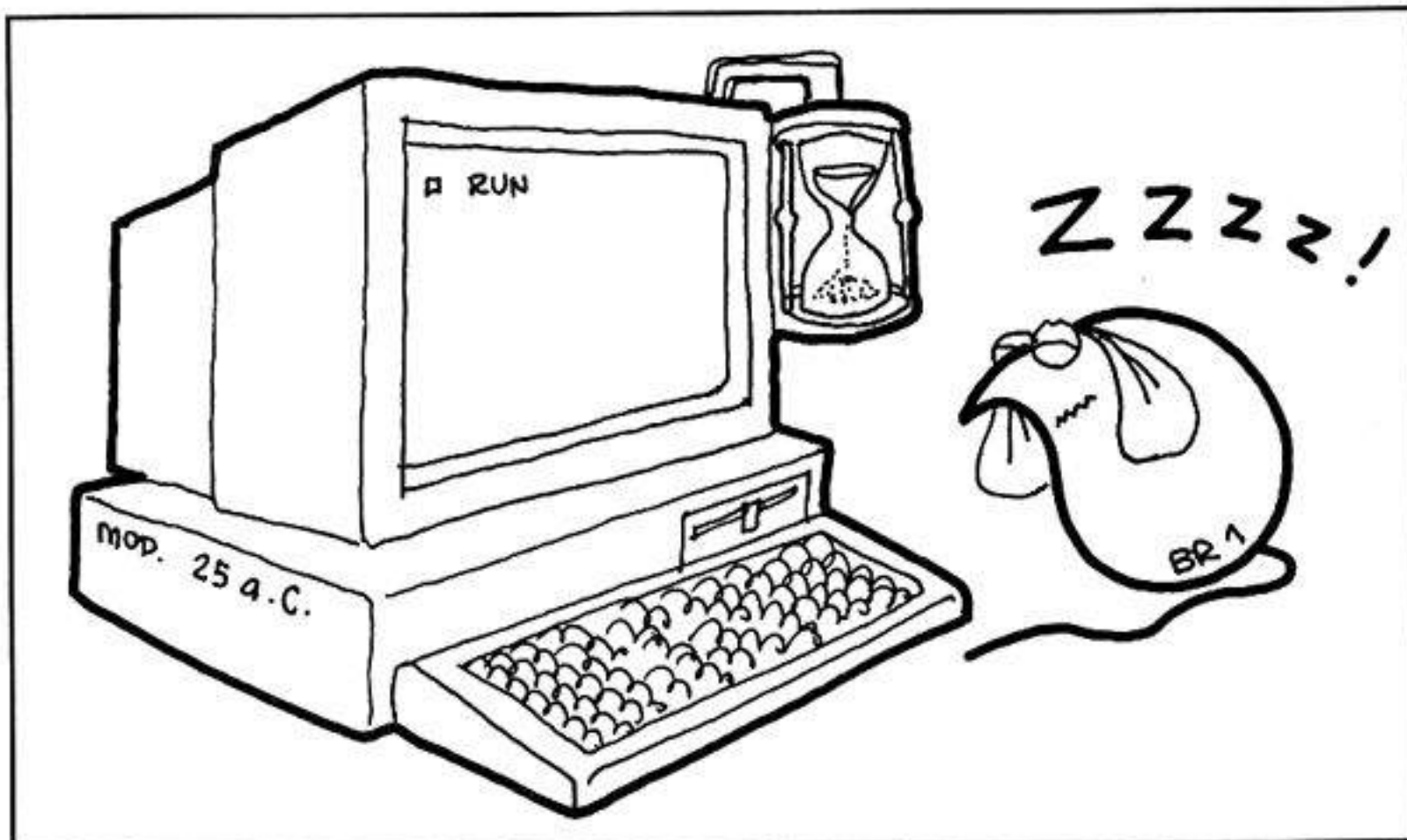
Inizialmente viene impostata la Ram come directory corrente (notare il comando **CD Ram:**); il motivo di tale artificio è di abbreviare i tempi di funzionamento della routine che, se fosse eseguita su un device lento come un normale drive, finché crea, concatena e cancella i vari files, potremmo sicuramente farci un pisolino. Se, però, non avete memoria da buttar via potrete sostituire la Ram con qualsiasi altro percorso, ad esempio la directory T (modificando, in questo caso, CD Ram: con **CD SYS:T**) ed eliminare il primo blocco di istruzioni e l'ultima riga dello Script (vedi **S**cript **F**ile **D**ecoder, **SCF** v1.01). In questo modo nella directory **T** verranno memorizzati i dati rilevati e aggiornati volta per volta.

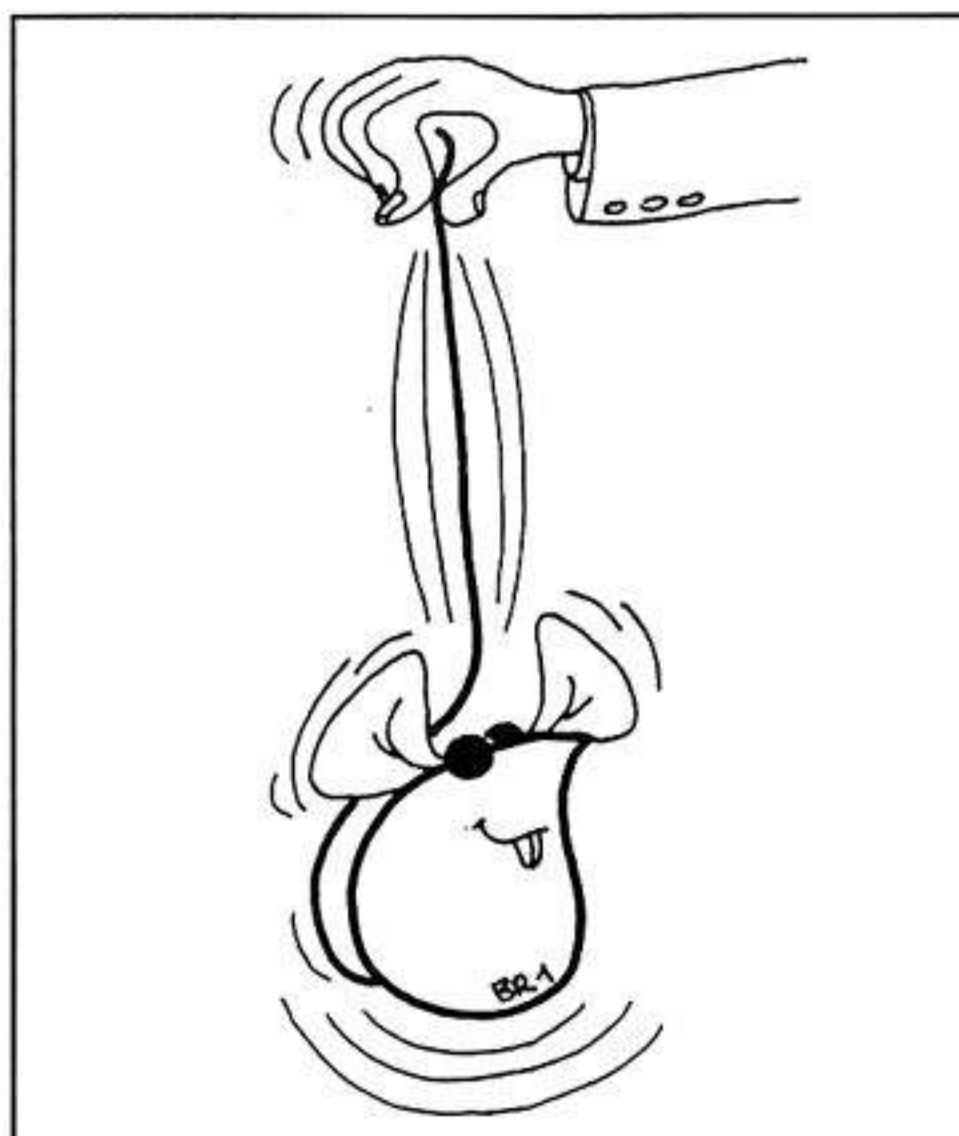
Lo Script si incarica, inizialmente, di appurare se esiste un file di nome **Data** in Ram; se lo trova, lavora su di esso, altrimenti lo cerca in T e lo copia in Ram. Se il file Data (destinato a contenere la lista degli orari) non esiste nè in Ram nè in T, viene inizialmente creato in Ram (memorizzandovi i primi due valori rilevati) copiandolo poi in T.

Per chiarezza, nei due file batch pubblicati sono stati inseriti i relativi commenti, istruzione per istruzione, in modo che la procedura possa essere compresa anche da chi è alle prime armi.

Precisiamo ancora che una delle due versioni è destinata a coloro che, possedendo un Amiga inespanso, non possono permettersi il lusso di "sprecare" Ram per rendere veloce la procedura; l'altra, invece, può esser digitata da chi non ha problemi di memoria.

Durante l'esecuzione della routine (qualunque delle due essa sia) vengono creati ben 5 (cinque!) file diversi; al termine, comunque, l'unico file che troverete è quello denominato **Data**, e che verrà situato nella directory **T**.

Una volta caricato il vostro editor preferito, anche il semplice **ED** del Dos, digitate le poche righe di una delle procedure pubblicate, avendo cura di inserire, al posto di **Nomeprogramma**, il nome ed il percorso completo del programma da lanciare. Memorizzate, poi, il tutto (in queste pagine ci riferiremo al nome SCF,

```
;                  ******************************************
;                  ***        SCRIPT-CLOCK-FILE  v1.0     ***
;                  ***                by                  ***
;                  ***     L a l l i   M i r k o  1991    ***
;                  ******************************************
;
CD RAM:                        ; Imposta la RAM-DISK come Courrent Directory
IF NOT EXISTS DATA             ; Se NON esiste il file DATA in RAM...
  IF EXISTS SYS:T/DATA         ; e se esiste tale file nella directory SYS:T...
    COPY SYS:T/DATA to RAM:    ; Copia tale file in RAM-DISK
  Endif
Endif

DATE >data1                    ; Legge la data iniziale la salva nel file data1
Nomeprogramma                  ; *** -> PROGRAMMA DA LANCIARE <- ***
DATE >data2                    ; Legge la data finale la salva nel file data2

JOIN data1 data2 to dataX      ; Concatena i due file con le date in dataX
DELETE QUIET data1|data2       ; Cancella i files originari

IF EXISTS DATA                 ; Se esiste il file DATA...
  COPY DATA to dataY           ; lo duplica in dataY...
  JOIN dataY dataX to DATA     ; e concatena dataX e dataY in DATA
Endif

IF NOT EXISTS DATA             ; Se il file DATA non esiste...
  COPY dataX to DATA           ; duplica dataX in DATA
Endif

DELETE QUIET dataX|dataY       ; Cancella i files superflui
COPY DATA to SYS:T/DATA        ; Copia il files DATA nella directory "SYS:T"
```

Il file batch per chi non ha problemi di memoria Ram

preso dalle iniziali di Script-Clock-File). Se per editare SCF non avete usato ED, ma un qualsiasi Word Processor, ricordatevi di salvarlo in puro formato Ascii.

Il file, a questo punto, è pronto per essere lanciato da CLI o Shell, semplicemente digitando...

`EXECUTE [path] SCF`

...in cui **path** è il percorso, cioè la directory dove volete memorizzare lo Script.

Per i più smaliziati non sarà difficile riuscire a svincolare, almeno apparentemente, il file dal CLI, e quindi mediante l'uso dell'istruzione **Iconx** renderlo operativo anche tramite icona da Workbench.

Visto che nessuno ha tempo da perdere, se avete abbastanza Ram rendete residenti i 7 comandi necessari al funzionamento dello Script; così facendo non vi accorgerete neppure della sua presenza, tanto è veloce.

A questo punto abbiamo ottenuto, nella directory T, un file che contiene una bella lista di date; che cosa ne facciamo? Così com'è memorizzato, è difficile riuscire a capire per quanto tempo abbiamo utilizzato questo benedetto programma: è quindi necessario un "qualcosa" che riesca ad interpretare la lista di date sommando gli orari in modo da fornirci il tanto sospirato *Tempo di Utilizzo*.

Il programma Basic

Questo *qualcosa* è un breve programma in Amigabasic che si incarica (quasi) automaticamente di fare tutti i calcoli e di comunicarci il responso.

```
;                    *****************************************
;                    ***      SCRIPT-CLOCK-FILE  v1.01      ***
;                    ***              by                   ***
;                    ***   L a l l i   M i r k o   1991    ***
;                    *****************************************
;
CD SYS:T                        ; Imposta 'T' come Courrent Directory

DATE >data1                     ; Legge la data iniziale la salva nel file data1
Nomeprogramma                   ; *** -> PROGRAMMA DA LANCIARE <- ***
DATE >data2                     ; Legge la data finale la salva nel file data2

JOIN data1 data2 to dataX       ; Concatena i due file con le date in dataX
DELETE QUIET data1|data2        ; Cancella i files originari

IF EXISTS DATA                  ; Se esiste il file DATA...
  COPY DATA to dataY            ; lo duplica in dataY...
  JOIN dataY dataX to DATA      ; e concatena dataX e dataY in DATA
Endif

IF NOT EXISTS DATA              ; Se il file DATA non esiste...
  COPY dataX to DATA            ; duplica dataX in DATA
Endif

DELETE QUIET dataX|dataY        ; Cancella i files superflui
```

Il file batch per Amiga inespanso

```
'        *********************************************
'        ***      SCRIPT-CLOCK-FILE   Decoder     ***
'        ***               by                     ***
'        ***         Lalli Mirko 1991             ***
'        ***                          for  C.C.C  ***
'        *********************************************

inizio:
REM  Inizio e Richiesta del nome del file
CLS
LOCATE 12,3
INPUT "Inserisci il nome del file dati (default = SYS:T/DATA)";file$
IF file$ = "" THEN file$ = "SYS:T/DATA"
PRINT "Apro il file indicato" : att : CLS

loop:
REM Determinazione numero di righe del file specificato
'ON ERROR GOTO CTRLerrori
OPEN "i",#1,file$
WHILE NOT EOF(1)
 a$ = INPUT$ (1,1)
 IF a$ = CHR$(10) THEN PRINT " -> ",a% : a% = a% + 1
 PRINT a$;
WEND
CLOSE
a% = a% -1
IF a% > 9 THEN DIM dati$(a%)
scrol (-11) : i% = 0 : CLS

loop2:
REM Lettura e memorizzazione dati da file specificato
OPEN "i",#1,file$
WHILE NOT EOF(1)
 a$ = INPUT$ (1,1)
 dati$(i%) = dati$(i%) + a$
 IF a$ = CHR$(10) THEN i% = i% + 1 : PRINT " -> ","OK!"
 PRINT a$;
WEND
CLOSE
scrol (-11) : CLS

convert:
REM Eliminazione parti data non necessarie
FOR i% = 0 TO a%
 dati$(i%) = RIGHT$(dati$(i%),9)
 dati$(i%) = LEFT$(dati$(i%),8)
 PRINT dati$(i%)
NEXT
PRINT : PRINT " Un attimo di pazienza ..."

calc:
REM Calcolo Tempi
FOR i% = 0 TO a% STEP 2
flg1% = 0
flg2% = 0
HH1 = VAL(LEFT$(dati$(i%),2))
```

Visto che sono necessarie operazioni di Input/Output sul dischetto, si è provveduto ad inserire una semplicissima routine di controllo degli (eventuali) errori.

Il listato è semplice e comprensibile anche per i neofiti dal momento che non si è fatto ricorso a librerie di alcun tipo.

Dopo aver dato il **Run**, il programma chiederà di inserire il **nome del file** da analizzare.

in accordo con la procedura pubblicata in queste pagine, tale file è: **SYS:T/Data** e, se non lo avete cambiato, premete Return senza inserire nulla.

A questo punto il file viene aperto, una *prima* volta, per rilevare il numero di righe, quindi di orari, in modo da dimensionare opportunamente una matrice che conterrà tutti i dati.

Alla *seconda* apertura del file, tutti gli orari verranno memorizzati e trattati per essere sommati. Eseguita anche questa operazione, il programma si incarica di dare il responso delle nostre fatiche.

Una piccola precisazione: il programma Basic (dal nome **SCFdecoder**) informerà anche del numero di *secondi*, oltre alle *ore* e *minuti*, pur se questo dato non può ritenersi completamente attendibile, in quanto l'orario iniziale viene registrato subito *prima* del caricamento del programma (i pochi secondi necessari al caricamento vengono conteggiati nel tempo di utilizzo); si tratta, però, solo di pochi secondi.

il listato Basic, inoltre, non viene appesantito da tale funzione che si è ritenuto utile implementare perchè la procedura di calcolo usata per ore, minuti, secondi, può essere utile anche per implementare, in un eventuale programma matematico, i **gradi sessagesimali**.

Lo script, e relativo programma, funzionano anche **se non avete un orologio interno** in quanto si limitano a considerare la *differenza di orario* rilevata all'inizio ed al termine dell'uso del programma. Ad esempio, la coppia di valori 12:45:30 (inizio) e 13:00:00 (fine) e la coppia 04:27:21 (inizio) e 04:42:51 (fine) sono equivalenti, perché comportano un tempo di utilizzo identico, pari a 15 minuti e 30 secondi.

Lo "scheletro" del programma realizzato si presta, come intuitivo, a miglioramenti funzionali ed estetici notevoli: l'occorrente per realizzarli è confinato, soprattutto, in una buona dose di fantasia e volontà.

```
MM1 = VAL(MID$(dati$(i%),4,2))
SS1 = VAL(RIGHT$(dati$(i%),2))
HH2 = VAL(LEFT$(dati$(i%+1),2))
MM2 = VAL(MID$(dati$(i%+1),4,2))
SS2 = VAL(RIGHT$(dati$(i%+1),2))
HHd = HH2 - HH1
MMd = MM2 - MM1 : IF MMd < 0 THEN MMd = MMd + 60 : HHd = HHd - 1
SSd = SS2 - SS1 : IF SSd < 0 THEN SSd = SSd + 60 : MMd = MMd - 1
SS = SS + SSd
 IF SS > 59 THEN SS = SS - 60 : flg1% = 1
MM = MM + MMd
 IF flg1% = 1 THEN MM = MM + 1
 IF MM > 59 THEN MM = MM - 60 : flg2% = 1
HH = HH + HHd
 IF flg2% = 1 THEN HH = HH + 1
NEXT

EXPOSE:
REM Esposizione DATI e conclusione
CLS
LOCATE 12,1 : PRINT " Il programma a cui appartiene il file dati ";file$
PRINT " h stato usato precisamente : " : PRINT
COLOR 3,0 : PRINT : PRINT HH;"  Ore"
PRINT MM;"  Minuti e"
PRINT SS;"  Secondi"
COLOR 2,0 : PRINT " Press any key ..."
WHILE INKEY$ = "" : WEND
END

CTRLerrori:
REM Routine controllo errori
CLS : BEEP
FOR K% = 1 TO 3
 LOCATE 15,20 : COLOR K%,0 : PRINT " A T T E N Z I O N E  !!!   Errore ."
 att : BEEP
NEXT
scrol (-12) : LOCATE 15,12
IF ERR = 53 THEN PRINT "Il file dati indicato non esiste !!!"
IF ERR = 57 THEN PRINT "Errore I/O Disco. 'Non ti resta che piangere!'..."
BEEP : att : att : scrol (-10) : COLOR 1,0 : GOTO inizio

'  Procedure

SUB att STATIC
 FOR att = 0 TO 3900 : NEXT att
END SUB : Ciclo di Ritardo

SUB scrol (ab%) STATIC
 FOR i = 1 TO 18
   SCROLL (0,1) - (615,256),0,ab%
 NEXT i
END SUB : Scrolling SCREEN
```

Il listato AmigaBasic, destinato ad "interpretare" il batch

di Luigi Callegari

La gestione delle stringhe in "ambiente" Amiga C

La manipolazione delle parole è uno dei principali problemi da risolvere in qualsiasi linguaggio di programmazione.

Il linguaggio **C** non prevede, tra i suoi tipi di base (vedi **figura 1**), il tipo **stringa**, come ad esempio il Basic.

Per manipolare parole, ovvero stringhe di caratteri ASCII, bisogna dunque ricorrere ai **puntatori** ed ai **vettori** di caratteri. Inoltre, le stringhe devono essere gestite tramite **funzioni di libreria** specifiche.

Ricordiamo, infatti, che le parole chiave del linguaggio C sono soltanto una trentina per lo standard ANSI (vedi **figura 2**). Tutto ciò che concerne la gestione delle stringhe e la conversione di caratteri (come anche i calcoli matematici, ad esempio) richiedono l'uso di funzioni "esterne" al linguaggio C.

Che cos'è una stringa

Una stringa viene rappresentata internamente da un compilatore C come una sequenza di caratteri ASCII, corrispondenti alle lettere che la compongono, terminate da un carattere di **NULL**, ovvero da un byte posto a zero.

Ad esempio, la stringa **Gina** viene memorizzata effettivamente come **Gina\0**, ovvero:

```
G   i   n   a   \0

047 105 110 141 000
```

In generale, il programmatore non deve preoccuparsi del *formato interno* di una stringa, se non per tenere conto che l'occupazione di memoria in numero di byte equivale alla **lunghezza della stringa più uno** (per il carattere NULL). Il mese scorso abbiamo parlato delle funzioni di **allocazione** della memoria; chi ci segue con regolarità ha già avuto un assaggio di come si gestisce la manipolazione della memoria in unione con le stringhe.

Vediamo ora alcune delle funzioni "standard" delle librerie di tutti i compilatori ANSI C (**Amiga** e **Ms - Dos**) e come potrebbero essere implementate dal programmatore usando delle istruzioni in linguaggio C.

Funzioni di libreria

Bisogna dire che, effettivamente, capita sovente che le funzioni di libreria del compilatore per la gestione delle stringhe vengano scritte in linguaggio C.

Nel caso di **SAS/C** ed **Aztec C**, però, si tratta di routine ottimizzate, scritte in linguaggio Assembler per la maggior parte, per garantire le minime dimensioni e la massima velocità operativa.

Le porzioni di codice assembly che svolgono tutte le funzioni delle librerie del compilatore sono inserite automaticamente dal programma linker (**BLink** per SAS/C, **LN** per Aztec C). Material-

Nome	Bytes	Val. Minimo	Val. Max
signed char	1	-128	+127
unsigned char	1	0	255
signed short	2	-32768	32767
unsigned short	2	0	65535
signed long	4	-2147483648	+2147483647
unsigned long	4	0	4294967295
float	4	+/-10E-37	+/- 10E+38
double	8	+/-10E-307	+/- 10E+308

Fig. 1. Tipi di dati del linguaggio SAS / C

```
auto       break      case
const      char       continue
default    do         double
else       emit       enum
extern     float      for
goto       if         int
long        register  return
short       signed    sizeof
static      struct    switch
typedef     union     unsigned
void        volatile  while
```

Fig. 2. Parole chiave ANSI

Il primo listato in C (Find); prima parte.

```
/* Programma Find.c per SAS/C V5.10 */
#include <string.h>
#include <ctype.h>
#include <stdio.h>
#define MAXSTRLEN   255/
* Restituisce la posizione nella stringa passata come primoargo-
mento del secondo argomento, oppure -1 se non trovata */
int indice( char linea[], char parola[] )
{
  int currentbase, lineindice, patternindice;
/* Verifica tutti i caratteri della stringa in sequenza */
for (currentbase=0 ; linea[currentbase]!='\0' ; currentbase++ )
 {
   for ( lineindice = currentbase, patternindice = 0 ;
   parola[ patternindice ] != '\0' &&
   parola[ patternindice ] == linea[ lineindice ];
   lineindice++, patternindice++ );
/* Il corpo del ciclo non ha istruzioni! */
/* Se tutti i caratteri di pattern corrispondono,si restituisce
currentbase, la posizione in linetocheckin cui è iniziata l'e-
guaglianza. */
   if( parola[ patternindice ] == '\0' )
       return( currentbase);
   }
   return( -1 );
}
/* Converte in minuscolo la stringa passata */
void lowerstring( char stringa[] )
{
   int i;
   for ( i = 0; stringa[ i ]; i++ )
      stringa[ i ] = tolower( stringa[ i ] );
}
/* Avanza di un caratt. alla volta sino alla fine della linea */
void stripnewlines( char stringa[] )
{
   int i;
   for ( i = 0 ; stringa[ i ] ; i++ )
      if ( stringa[ i ] == '\n' )
```

mente sono porzioni di codice in linguaggio macchina 68000 archiviate in un formato particolare all'interno del file **LC.LIB** per SAS/C e **C.LIB** per Aztec C. Le funzioni matematiche sono memorizzate in analoghi file di libreria di linking separati da questi, scandite dal linker quando deve costruire un file eseguibile (quindi completo di tutte le routine e non solo di quelle generate in base alle istruzioni del listato sorgente scritto dal programmatore) partendo dal modulo oggetto generato dal compilatore (**LC** per Lattice o **CC** ed **AS** per Aztec).

Il linker usa anche altre librerie per creare un programma eseguibile da Shell, come ad esempio **Amiga.lib** che contiene tutto il codice di interfacciamento necessario ai programmi per usare le funzioni nelle librerie di sistema di Amiga (Exec, Graphics, Intuition, Layers...). Ritorneremo in futuro, quando saremo un poco più esperti, sui dettagli di funzionamento delle librerie dei compilatori C.

Lunghezza di stringa

La funzione di libreria standard che calcola quanti caratteri sono contenuti in una stringa C si chiama **strlen()**.

Assume come unico parametro un **puntatore alla stringa** interessata e restituisce, come risultato, un numero intero pari alla **lunghezza** in caratteri della stringa, contati dall'inizio alla chiusura (marcata col byte di NULL, come detto poc'anzi). Ad esempio:

```
void main()
{
```

```
        {
            stringa[ i ] = '\0';
            break;
        }
}
void main()
{
  char lineacorre[ MAXSTRLEN ], filename[ MAXSTRLEN ];
  FILE *infile;
  char searchstring[ MAXSTRLEN ];
  /* Apri il file di input */
  printf("Immetti il nome del file da cercare: ");
  fgets( filename, MAXSTRLEN, stdin );
  stripnewlines( filename );
    if ( ! ( infile = fopen( filename, "r" ) ) )
  {
      printf("Non si apre il file di input!\n");
      exit( 0 );
  }
  /* Leggi la stringa da ricercare */
  printf( "Immetti la stringa da cercare: ");
  fgets( searchstring, MAXSTRLEN, stdin );
  /* Elimina i caratt. di fine linea e converte in maiuscolo */
  stripnewlines( searchstring );
  lowerstring( searchstring );
  /* Legge il file una linea per volta */
  while( fgets( lineacorre, MAXSTRLEN, infile ) != 0 )
  {
      lowerstring( lineacorre );
       /* Se si trova la stringa, si stampa la linea */
    if ( indice( lineacorre, searchstring ) != -1 )
      fputs( lineacorre, stdout );
  }
}
/* Fine del programma */
```

Il primo listato in C (Find); continuazione e fine.

Il secondo listato (Crypt); prima parte.

```
/* Programma Crypt.c per SAS/C V5.10 */
#include <string.h>
#include <ctype.h>
#include <stdio.h>

#define  EQUALS      0
#define  MAXSTRLEN  255
#define  PASSWORD    "teresa"
/* Converte in minuscolo la stringa passata */
void lowerstring( char stringa[] )
{
   int i;
   for ( i = 0; stringa[ i ]; i++ )
      stringa[ i ] = tolower( stringa[ i ] );
}
```

```
    int n;

    char *s = "Cristina";
    n = strlen( s );
    printf( "%d\n", n );
}
```

In questo caso, verrà stampato "**8**", ovvero il numero di caratteri inserito nella variabile intera "**n**" dalla funzione **strlen()**.

Per usare le funzioni di gestione delle stringhe è buona norma includere, tramite l'apposita direttiva di preprocessor, il file "**string.h**":

```
#include <string.h>
```

Vediamo ora come si potrebbe scrivere la funzione strlen() in linguaggio C, supponendo che non sia presente nella libreria fornita col compilatore. O, con meno fantasia, per capire come funziona:

```
int strlen( char *s )
{
int lungo = 0;

while( *( s + lungo ) != '\0' )
++lungo;
return( lungo );
}
```

In entrata, viene passato l'indirizzo dell'inizio della stringa che viene assegnato alla variabile puntatrice "**s**". La variabile **lungo** viene definita come intera e azzerata per consentire il conteggio dei caratteri che intercorrono tra l'inizio della stringa (**s**) ed il carattere NULL, che dobbiamo cercare. Per farlo, usiamo l'operatore asterisco (*), che restituisce il carattere contenuto all'indirizzo specificato dal valore dell'espressione che segue (s+lungo). Nel nostro caso, viene restituito il carattere collocato all'**indirizzo di partenza** della stringa da verificare, sommato alla **lunghezza** calcolata progressivamente ed incrementata (`++lungo`) sinchè (`while`) non viene trovato il byte '\0'. Si usa ovviamente la keyword `return()` per restituire al programma chiamante il contenuto della variabile "`lungo`".

Per la similitudine tra puntatori e vettori, la funzione si sarebbe potuta scrivere anche con altri sistemi, come vedremo con le prossime funzioni di base.

```
/* Avanza di un caratt. alla volta sino alla fine della linea */
void stripline( char stringa[] )
{
   int i;
   for ( i = 0 ; stringa[ i ] ; i++ )
      if ( stringa[ i ] == '\n' )
{
         stringa[ i ] = '\0';
         break;
   }
}
void main()
{
   int i, plenght;
   char temp, upassword[ MAXSTRLEN ];
/* Qui l'utente digita la sua parola d'ordine */
   printf("Immetti password, poi premi Return: \n");
   fgets( upassword, MAXSTRLEN, stdin );
/* Elimina i caratteri di fine linea e le maiuscole */
   stripline( upassword );
   lowerstring( upassword );
/* Determina la lunghezza della tringa utente */
   plenght = strlen( upassword );
/* Esegue codifica scambiando i-esimo caratt.con plenght-i-1 */
   for ( i = 0 ; i < plenght ; i++ )
   {
      temp = upassword[ i ];
      upassword[ i ] = upassword[ plenght - i - 1 ];
      upassword[ plenght - i - 1 ] = temp;
   }
   if ( strcmp( upassword, PASSWORD ) != EQUALS )
   {
      printf("La tua parola d'ordine _ errata!\n");
      exit( 0 );
   }
   else
   {
      printf("La tua parola d'ordine _ corretta.\n");
   }
}
/* Fine del programma */
```

Il secondo listato (Crypt); continuazione e fine.

Confronto di stringhe

Nelle librerie dei compilatori esistono diverse funzioni di comparazione di stringhe.

La più comune è comunque la **strcmp()**, che accetta come parametri i due puntatori alle stringhe da confrontare, restituendo un valore pari a **zero** (se le stringhe sono identiche), **positivo** (se la prima è superiore alla seconda) o **negativo** (in caso contrario). Spesso, ma non sempre, il valore restituito è la differenza tra i primi due caratteri ASCII trovati diseguali nelle stringhe. Vediamo un esempio:

```
void main()
{
char a[30], b[30];

printf("Immetti primo nome ");
scanf( "%29s", &a0 );

printf("Immetti secondo nome ");
scanf( "%29s", &b[0] );

if ( strcmp( a, b ) == 0 )
printf("I nomi sono eguali!\n" );

else if ( strcmp( a, b ) > 0 )
printf( "%s > %s\n", a, b );

else
printf( "%s < %s\n", a, b );
}
```

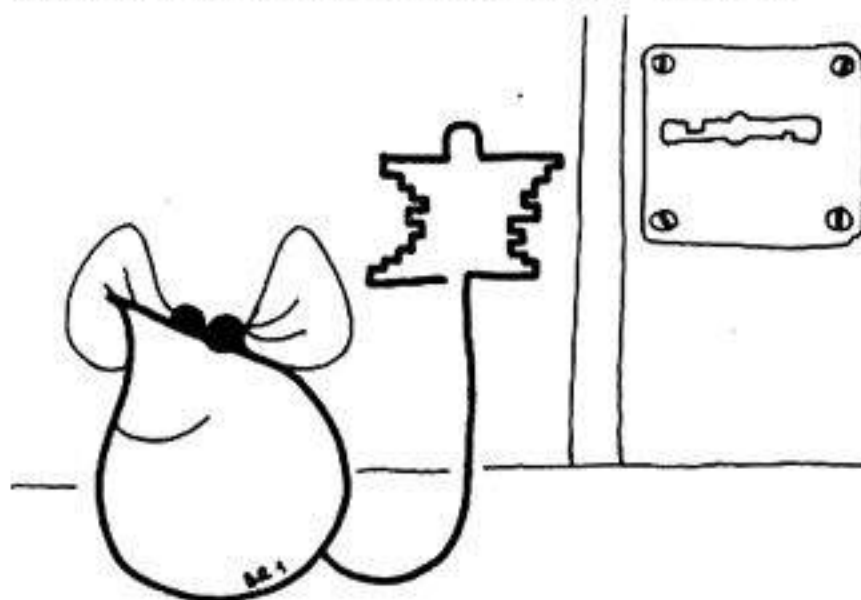
Facendo delle prove con questo programma si noterà che la funzione **strcmp()** considera differenti i caratteri maiuscoli e minuscoli, ovvero è "case sensitive" per dirla all'inglese. Ciò significa che la stringa "Papa" e la stringa "PaPa" sono viste come differenti, e la prima è considerata **maggiore** della seconda perchè i caratteri ASCII maiuscoli hanno valori numerici inferiori di quelli minuscoli.

Vediamo ora come scrivere la funzione **strcmp()** in linguaggio C:

```
int strcmp(char *a,char *b)
{
for (;*a == *b;a++;b++)
{
   if (*a == '\0')
      return( 0 );
   }
return((int)*a-*b);
}
```

In questo caso restituiamo un valore pari alla differenza dei primi due caratteri differenti nella stessa posizione relativa all'inizio di ciascuna delle due stringhe.

Copia di stringhe

Dal momento che nel linguaggio C non esiste il tipo di dato "stringa", non è consentito copiare, come in Basic ad esempio, una stringa entro un'altra

con una semplice assegnazione (**a$ = b$**). Bisogna usare una funzione di libreria, ad esempio **strcpy()**, che esegua materialmente la copia dei byte rappresentanti i caratteri ASCII di una stringa nelle locazioni di memoria rappresentanti un'altra stringa (di destinazione).

I parametri di *strcpy()*, come per *strcmp()*, sono dunque due puntatori all'**inizio** delle stringhe: la prima è dove viene copiata la stringa, la seconda è quella che contiene la stringa da copiare. Il contenuto delle locazioni di memoria della prima stringa sono sovrascritti e perduti; quelli della seconda restano intatti. Ad esempio:

```
void main()
{
char a[30];
char *b = "Commodore";
strcpy( a, b );
printf( "a: %s, b:
%s\n", a, b );
}
```

Si noterà che la stringa "**a**" ha assunto il contenuto della stringa "**b**".

Si noti che funzioni come strcpy() non effettuano verifiche sulle dimensioni dei vettori in gioco: ciò significa che, copiando in un vettore, ad esempio, definito come richiedente uno spazio in memoria per 20 caratteri una stringa lunga 30 caratteri, il compilatore non segnalerà alcun errore, ma il programma finale funzionerà sicuramente male (e chiamerà probabilmente il **Guru**). Esistono nelle librerie standard, comunque, funzioni come **strncpy()** che eseguono la copia sino ad un dato numero di caratteri (rappresentato da un numero intero, terzo parametro).

Vediamo come sia facile e veloce scrivere in linguaggio C una versione personale della funzione:

```
void strcpy( char *a, char *b )
{
    while( *a++ = *b++ )
        ;
}
```

in pratica, viene eseguita la assegnazione (=) nella locazione designata dalla variabile "**a**" di quanto proveniente dalla locazione indicata dal puntatore "**b**", ed ambedue vengono incrementati dopo l'operazione di assegnazione con un operatore di postincremento. L'incremento (**++**) di un puntatore ha l'effetto di portarlo all'indirizzo del carattere successivo. Il valore copiato funge anche da espressione logica restituita a **while**: sinchè è differente da zero, *while* assume che la condizione sia vera e prosegue l'assegnazione e gli incrementi di indirizzi, dato che il corpo del ciclo while è l'istruzione nulla (;). Dal momento che la funzione strcpy() non restituisce alcun valore, viene dichiarata come void.

stpblk()	Salta gli spazi vuoti in una stringa
stpbrk	Trova il carattere di separazione in una stringa
stpchr	Trova un carattere in una stringa
stpchrn	Trova un carattere non presente in una stringa
stpcpy	Copia una stringa in un'altra
stpdate	Converte un array di data in stringa
stpsym	Legge il simbolo successivo da una stringa
stptime	Converte un array di orario in un astringa
stptok	Legge un token da una stringa
stbpl	Costruisce una lista di puntatori di stringa
strcat	Concatena stringhe
strchr	Trova un carattere in una stringa
strcmp	Confronta due stringhe
strcmpi	Confronta due stringhe (case insensitive)
stcpy	Copia una stringa su di un'altra
strcspn	Misura l'ampiezza di caratteri non in un set
strdup	Duplica una stringa
stricmp	Confronta due stringhe (case insensitive)
strins	Inserisce una stringa
strlen	Calcola la lunghezza di una stringa
strlwr	Converte una stringa in minuscolo
strmfe	Crea un nome di file con estensione
strmfn	Crea un nome di file con componenti
strmfp	Crea un nome di file con path/nodo
strmid	Restituisce la sottostringa di una stringa
strncat	Concatena stringhe, massima lunghezza
strncmp	Confronta stringhe, a lunghezza limitata
strncpy	Copia stringa, a lunghezza limitata
strnicmp	Confronta stringhe, case insensitive, limitata
strnset	Assegna un valore ad una stringa
strpbrk	Trova un carattere di interruzione nella stringa
strrchr	Trova un carattere non nella stringa
strset	Assegna un valore ad una stringa
strsfn	Spezza il nome del file
strspn	Misura l'ampiezza dei caratteri in un set
strtok	Legge un token
strtol	Converte una stringa in long
strupr	Converte una stringa in maiuscolo
stspfp	Scandisce la path del file

Fig.3. Funzioni di gestione stringhe in SAS / C

Concatenazione

Per inserire una stringa in coda ad un'altra già esistente bisogna innanzitutto assicurarsi, durante la stesura del listato sorgente, che il vettore di locazioni di memoria riservate alla variabile di destinazione sia sufficiente a contenere il risultato, al momento dell'esecuzione.

La funzione di libreria standard che concatena due stringhe è **strcat()**, che prevede come parametri i soliti due puntatori a stringhe ed inserisce la seconda a partire dalla coda della prima, terminandola con un NULL. Vediamo un esempio:

```
void main()
{
   char a[40], b[20];

printf("Nome? ");
scanf("%19s", &a[0];
printf("Cognome? ");
scanf("%19s", &b[0]);
strcat( a,b );
printf("Sei %s \n!", a
);
}
```

Si noti che la concatenazione della stringa del cognome viene fatta in quella del nome, che infatti è dimensionata inizialmente per poterle contenere entrambe.

Volendo scrivere la funzione **strcat()** senza usarne altre di quelle già viste, si potrebbe scrivere:

```
void strcat( char *a, char *b )
{
```

```
    while( *a != '\0' )
        ++a;
    while( *a++ = *b++ )
        ;
}
```

...dove il primo ciclo **while** colloca il puntatore "**a**" sul NULL di chiusura della prima stringa, mentre il secondo ciclo ricopia semplicemente la seconda stringa nella prima, sinchè non viene incontrato il NULL.

Volendo usare le funzioni di gestione stringhe già viste, si potrebbe usare molto più semplicemente...

```
void strcat( char *a, char *b )
{
strcpy( a + strlen( a ), b );
}
```

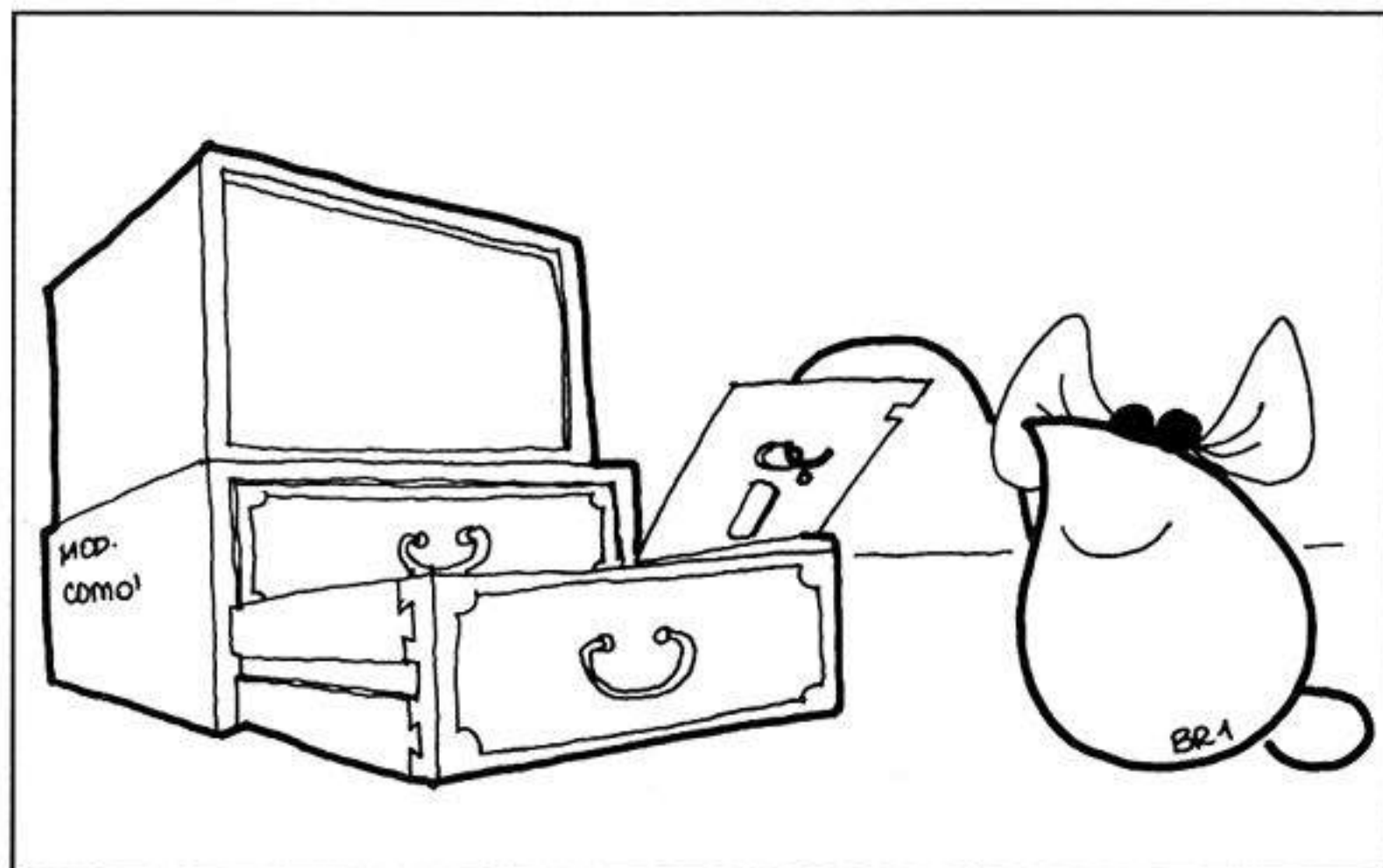
Verifica se il carattere è...	
isalnum	alfanumerico
isalpha	alfabetico
isascii	ASCII
iscntrl	di controllo
iscsym	simbolo C
iscsymf	simbolo C di testa
isdigit	cifra decimale
isgraph	carattere grafico
islower	minuscolo
isprint	stampabile
ispunct	di punteggiatura
isspace	di spaziatura
isupper	maiuscolo
isxdigit	esadecimale

Fig. 4. Funzioni di verif. caratt. in SAS / C

...assegnando come indirizzo di inizio della memorizzazione fatta da strcpy() la somma tra il puntatore di inizio della prima stringa e la sua lunghezza. Così si ottiene infatti l'indirizzo della locazione che ne contiene il NULL di terminazione, da dove partirà la copiatura dei caratteri letti da "**b**".

Programmi

Riportiamo, in figura il listato del programma **Crypt.c**, piuttosto semplice, che può perciò illustrare chiaramente il funzionamento di alcuni dei criteri di manipolazione delle stringhe.

Il programma parte chiedendo di immettere una **parola d'ordine**, poi la decodifica e confronta il risultato con la parola d'ordine definita (**#define**) in testa al programma compilato ed infine visualizza se le password corrispondono o meno.

L'algoritmo di crittografazione è semplice: si invertono le posizioni di tutti i caratteri nella stringa, con un semplice ciclo **for**.

Il secondo programma di cui presentiamo il listato commentato svolge una funzione leggermente più evoluta: ricerca, in un file ASCII specificato, tutte le linee che contengono una certa parola. Si può qui studiare ancora il funzionamento delle funzioni di base per l'apertura e la lettura di file, già viste due numeri fa, nonchè le funzioni di manipolazione di stringhe.

Ricordiamo che per compilare e linkare in un file eseguibile un programma con **Lattice C** si deve usare semplicemente...

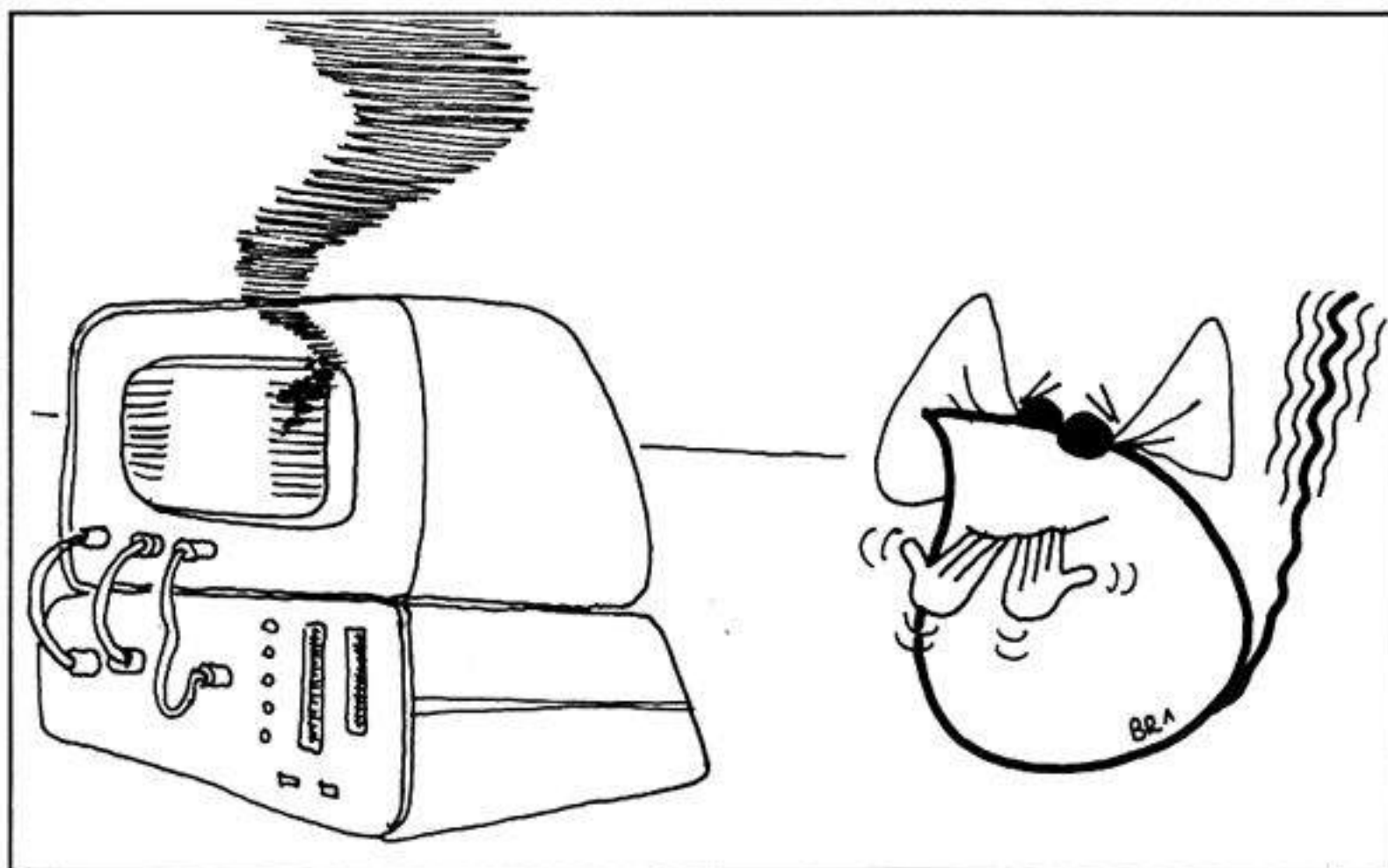
```
LC -L nomep.c
```

...mentre con **Aztec C** sono necessari due comandi da Shell:

```
CC Nomep.c
LN Nomep.c -lc
```

Per eseguire i programmi basta poi digitarne il nome da Shell.

POSTAMIGA

(a cura di Domenico Pavone)

Spazi... ingiustificati

Adopero molto il mio Amiga 500 per stilare testi, sfruttando il W/P C1-Text 3.0. Non riesco, però, ad incolonnare a video (non ho la stampante) quanto scrivo senza lasciare spazi a destra, come per esempio vedo nella stampa delle riviste. Ho provato ad impostare da menu la sillabazione a fine riga, ma senza alcun risultato. Inoltre non so creare più di una colonna, per ottenere un effetto come quello dei quotidiani (o della stessa Postamiga). Saluti da un vostro vecchio lettore, ho 47 anni....
(A. Martinucci - Savona)

L'allineamento del testo sulla destra, detto anche justify, o meglio **giustificazione a destra** visto che si sta parlando dell'italianissimo C1_text, è di fatto implementato dal word processor, ma finalizzato alla (eventuale) successiva fase di stampa.

In altre parole, non è possibile vedere questo tipo di output direttamente sullo schermo, almeno in fase di editing. E' opportuno precisare, tra l'altro, che l'opzione di allineamento non è di per sé legata alla sillabazione a fine riga, che può, tutt'al più, compattare ulteriormente il testo, senza però eseguire una vera giustificazione dello stesso. Per ottenerla, è necessario selezionare dal menu **Parametri / Impaginazione / Paragrafo** l'opzione **Allineamento Bilaterale** (oppure **Destra**, ma in questo caso perdendo l'allineamento alla sinistra). Impostando a *Sì* la sillabazione, si avrà tanto l'allineamento che l'automatico *a capo* con corretta troncatura dei vocaboli, ottenendo un effetto di maggiore compattezza del testo.

Per sfruttare visivamente l'allineamento senza disporre di una stampante, si può salvare il testo tramite l'opzione **Memorizzare Testo / Stampa Ascii**. Si provi, dopo aver salvato un documento in questa modalità, a ricaricarlo in ambiente C1-Text: si potrà toccar con mano l'avvenuto allineamento, che comporterà in pratica l'aggiunta di un carattere **EOL**, ovvero *fine linea* (che C1-Text, normalmente, visualizza). Chiaro che, sempre presupponendo il mancato uso di una stampante (in verità un po' blasfemo), la memorizzazione come "stampa ascii" trova un impiego pratico se poi si pensa di visualizzare il testo in ambiente dos (da Shell oppure Cli, per intendersi) con qualcosa come *Type Nomefile*, oppure sfruttando programmi tipo *More*. In entrambi i casi, quanto prodotto dal C1-text verrà mostrato perfettamente allineato. Attenzione, però: adoperando l'opzione *Salva Ascii* verrebbero estromessi eventuali stili particolari come il neretto, il sottolineato, eccetera. Qualora fossero presenti nel testo simili caratteristiche, per mantenerne attiva la visualizzazione anche con un banale Type (oppure More) si renderebbe necessario ricorrere a **Stampa Ansi**, sempre dal menu **Generale / Memorizzare Documento** di C1-Text.

Quanto alla rappresentazione del testo in più colonne (per dirla sullo stile del nostro lettore), questa **non** può essere implementata da un semplice text editor. Simili prodezze sono riservate a programmi di Desk Top Publi-

Basic e sfondo

Ho cominciato da poco a programmare in Basic. Vorrei sapere se si può cambiare il colore di sfondo di una finestra per poi disegnare con i comandi Circle, Box, eccetera, e se si può disegnare con un tratto più spesso senza manovre troppo complicate.
(Mario Capria - Napoli)

Certo che si può, ed è anche piuttosto facile. Per ottenere uno sfondo omogeneo di diverso colore basta adoperare opportunamente il comando **Color**, e poi farlo seguire da un **Cls** per pulire lo schermo.

Se, per esempio, si intende avere tutta la finestra corrente con sfondo arancio, sempre che non si sia modificata la Palette generale, basterà inserire nel listato una sequenza **Color 1, 3 : Cls** (la si può anche provare in modalità diretta, digitandola nella finestra di output). L'intera finestra assumerà il colore **3** (per default corrispondente al rosso-arancio) per lo sfondo, mentre eventuali tracciamenti avverrano in bianco (il colore **1**). Per evitare che questa impostazione resti attiva anche dopo l'uscita dal programma, è opportuno **prima** della fine settare nuovamente i colori di default, ovvero adoperare **Color 1, 0: Cls**. Come ovvio, queste ultime istruzioni diventano superflue se si adoperano schermi e finestre diversi da quella di output del basic (si veda l'esempio riportato).

Dopo l'assunzione del nuovo colore di sfondo, potrà essere adoperata qualunque istruzione di tracciamento grafico... ma non certo Box. E per un motivo molto semplice: AmigaBasic non comprende una simile istruzione (il lettore proviene evidentemente da altre esperienze computerecce). Per disegnare riquadri occorre sfruttare il comando **Line**, come peraltro ben illustrato dal manuale.

Quanto allo **spessore** del tratto, a meno di non ricorrere al complicato Pattern che richiede una certa conoscenza di bit, byte ed amenità simili, si può sfruttare un banale artifizio: disegnare più figure, leggermente "sfalsate" una rispetto all'altra. Più facile a farsi che a dirsi, si provi a digitare e mandare in esecuzione questo breve listato:

```
SCREEN 2, 640, 250, 2, 2
WINDOW 2, , ,0,2
COLOR 1,3:CLS
spessore=5
FOR x=1 TO spessore
  CIRCLE (320,125),80+x
NEXT
PRINT "CLICKA COL MOUSE"
WHILE MOUSE(0)=0:WEND
WINDOW CLOSE 2
SCREEN CLOSE 2
END
```

Si vedrà applicato quanto detto a proposito della colorazione dello sfondo, nonché della simulazione di un maggiore spessore del tratto, la cui entità dipende dalla variabile **spessore**. Si provi ad assegnarle valori diversi, evitando lo zero (che non produrrà alcun disegno) o valori troppo alti, più che altro per l'inevitabile rallentamento legato al tracciamento di troppe figure.

Come si può notare, un semplice ciclo **For...Next** traccerà tante figure quante specificate dalla variabile *spessore*, in questo caso dei cerchi, ma la stessa tecnica può essere adoperata per riquadri, o semplici linee. Si badi anche alla risoluzione video adottata dall'istruzione **Screen**: adottando la 320 x 256 (low), l'effetto risulta meno marcato. Anche in questo caso, l'invito è a sperimentarlo concretamente, basterà sostituire l'istruzione del listato con **Screen 2, 320, 250, 2, 1**.

Posta "Principianti"

shing (**DTP**) quali **Professional Page, Page Stream** e similari, in grado di produrre vere impaginazioni nel formato che più aggrada, compreso quello su più colonne come le pagine di questa rubrica.

Avventurarsi in questi settori senza disporre di una stampante, è però piuttosto incoerente... anche per un "vecchio" lettore di CCC.

Che comunque non deve pensare di essere il solo in età non più verdissima. Pensi che tutti i collaboratori di questa rivista, a turno, si danno quotidianamente da fare per imboccare (di chip) il loro megadirettore, che ha ormai abbondantemente varcato la soglia del centenario. (*...questa me la paghi; ndr...*)

Saluti da un... ex... collaboratore.

Dei massimi sistemi

Ultimamente mi ha lasciato perplesso la decisione della rivista di dedicare grande spazio al mondo Ms-Dos, per il quale esistono già riviste specifiche, ma la goccia che ha fatto traboccare il vaso è un'affermazione apparsa sul n. 80, in cui viene dimostrato che l'Ms-Dos ha definitivamente "sconfitto" l'Amiga... (omissis) ...Mi chiedo che cosa abbia indotto una rivista indirizzata ad utenti di sistemi Commodore ad affermazioni simili...
(Massimo Pagani - via BBS)

Precisiamo che questa è solo una sintesi della lettera del nostro lettore, come molte altre troppo lunga per essere pubblicata integralmente. Il problema di cuore di un amigo è comunque palpabile, e merita una risposta, anche se forse non potrà soddisfarlo appieno.

Intanto, cominciando dalla coda della missiva, è da non dimenticare che la **Commodore** non produce soltanto **Amiga**, o soltanto il vecchio C/64. Anzi, la serie di **Pc compatibili** commercializzata col marchio Commodore abbraccia tutta la gamma che va dai "piccoli" XT 8088 agli AT 80286, 80386 e (imminenti o già in circolazione quando si leggeranno queste righe) 80486.

Cade così una delle obiezioni poste.

Per il resto, e queste righe sono scritte da un profondo estimatore di Amiga, è giusto che la nostra rivista dia spazio anche a chi preferisce per un motivo o per l'altro i sistemi Ms - Dos. Per i quali, è vero,

```
LIBRARY "graphics.library"
WIDTH 60
FOR x=1 TO 300
 PRINT "- - ";
NEXT
rp&=WINDOW(8)
WHILE MOUSE(0)<>-1:WEND
x1&=MOUSE(3):y1&=MOUSE(4)
setdrmd& rp&,2
loop:
x2&=MOUSE(1):y2&=MOUSE(2)
Rectfill& rp&,x1&,y1&,x2&,y2&
WHILE MOUSE(0)=-1
 IF x2&<>MOUSE(1) OR y2&<>MOUSE(2) THEN
 Rectfill& rp&,x1&,y1&,x2&,y2&
 GOTO loop
 END IF
WEND
setdrmd& rp&,1
LIBRARY CLOSE
END
```

Listato per selezionare una parte di testo

esistono anche altre pubblicazioni, la stessa Systems editoriale ne produce (p. es. **Personal Computer**). Ma, come in passato avveniva per computer oggi superati, quante di queste si rivolgono ad una utenza con tanta voglia di "smanettare", ma che ancora sta muovendo i primi e più difficili passi nell'ardua impresa di conoscere la macchina che gli sta davanti?

Detto questo, c'è anche da considerare che non esiste un sistema migliore o peggiore in assoluto, ma solo in rapporto all'uso che si intende farne. Amiga, inutile dirlo, è insuperabile nelle applicazioni grafiche, sonore e ludiche (perché sottovalutare i games?), ma anche in molti altri insospettabili (per i non addetti) settori, come il Dtp o il Dtv. Tuttavia, in molti casi può essere anche giusto considerare le esigenze di chi si rivolge ad altri sistemi, Ms - Dos in primis. Vetusto, talvolta irritante se si pensa alla facilità con la quale Amiga risolve certi problemi nella sua versione più elementare (leggi: 68000, senza dover ricorrere al già circolante 68030), eppure (non lo si dimentichi) con una attenzione non disprezzabile a chi il Pc "l'aveva già", e che trova sempre una compatibilità possibile con i programmi di quel mondo che circolano anche a distanza di anni dall'evoluzione del computer.

Tra l'altro, CCC non è mai stata una rivista dedicata esclusivamente ad un computer, tantomeno ad Amiga.

Noi "amighi" abbiamo per così dire il palato fine, ma questo non deve significare chiudersi nella proverbiale torre d'avorio...

Telegrafica

E' possibile adattare, in modo relativamente semplice ed economico, la comune espansione di memoria da 512K per Amiga 500 ad un Amiga 2000?
(Luca Bertin - Padova)

No.

Buon lavoro

Vi invio un disco contenente una beta version del mio futuro copiatore, che ho chiamato BitJoker Power Copy, anche per porvi alcune domande su problemi che non ho ancora risolto...
(Claudio Noventa - Padova)

Non ci dilunghiamo sui quesiti per una basilare informazione che il lettore ha dimenticato di fornire: il linguaggio di programmazione adottato. Tra l'altro, senza una direttiva di massima sugli algoritmi seguiti, non è semplice poter dare dei consigli, quale che sia il tema. Volendo azzardare un giudizio, di lavoro da fare comunque ce n'è tanto, l'interfaccia presenta ancora parecchi problemi, e ci si consenta una osservazione: pur se può essere utile per approfondire la conoscenza di Amiga, da un punto di vista pratico, un copiatore tutto sommato "normale" (anche se in evoluzione) si inserisce in un settore già notevolmente ricco di scelte, ed a livelli piuttosto alti.

Non resta quindi (per ora) che rivolgere al lettore di Padova un augurio di buon lavoro, e una benevola "tiratina d'orecchi" per la sua involontaria (almeno ci auguriamo) disattenzione: il floppy inviatoci era contagiato dal virus Lamer Exterminator!

Quali testi?

Sono un appassionato di Basic sin dai tempi del C/64, ma non riesco a trovare un buon libro sul Basic di Amiga, soprattutto che mi spieghi bene l'uso delle librerie di sistema. Ne ho comprato uno, ma vi ho trovato solo miseri esempi, e delle librerie neanche un accenno. Esiste qualcosa del genere?
(Giorgio Orsi - Livorno)

Di libri dedicati al basic ne circolano parecchi, ma è difficile trovarne qualcuno che tratti solo l'uso delle librerie di sistema, e per un motivo tutto sommato comprensibile: sono praticamente una marea, e per di più la stragrande maggioranza non sfruttabili da questo linguaggio.

Esistono comunque testi che dedicano un più o meno nutrito numero di pagine all'argomento, come ad esempio **Tricks & Tips** della Data Becker (non riservato solo al basic), o **Advanced Amiga Basic**, questo però in lingua inglese, edito dalla Compute Editions.

Notepad? no, grazie

Ho da poco un modem, e mi sono collegato alla vostra Bbs più di una volta. Mi è successa, però, una cosa strana: volevo lasciare un messaggio al mitico Red per dei chiarimenti, così ho pensato di scriverlo prima col Notepad; in seguito l'ho spedito come file di testo (uso il programma Jrcomm per Amiga). Risultato: tutto è andato in tilt, costringendomi a staccare la linea. Forse si deve per forza scrivere il messaggio mentre si è collegati?
(Pietro Servidio - Varese)

Il motivo dell'inconveniente è presto spiegato: un normale messaggio, e questo vale per qualunque **bbs**, deve essere fisicamente rappresentato da un testo strettamente in **Ascii**, e quindi per scriverlo è indispensabile adoperare un **editor** che consenta una memorizzazione in puro Ascii.

Opzione, questa, implementata un po' da tutti i word processor e text editor: **C1- Text**, Prowrite, CygnusEd, eccetera.

Notepad, invece, non rientra in questa categoria perchè assieme al testo vero e proprio vengono salvati, nel file prodotto, anche caratteri di controllo del formato. Ovvia deduzione, Notepad non va **mai** adoperato per scrivere testi da inviare ad una bbs, così come è da evitare (tassativamente) un suo uso per stilare programmi in basic, o file batch per il Dos.

Nel caso dell'invio dell'abortito messaggio via modem, quei caratteri non Ascii hanno probabilmente *ingannato* la bbs, che li ha interpretati secondo un suo "personale" criterio di giudizio, innescando chissà quali strani comportamenti. Anche adoperando un corretto text editor, si tenga presente un'altra possibile fonte di malfunzionamenti: le **righe vuote**. Queste, in pratica, sono normalmente rappresentate da un solo carattere, il return. Tale situazione, in molti programmi di bbs (incluso quello adoperato da **bbsystems**) indica la conclusione dell'editing, per cui si può anche rischiare che la ricezione del messaggio venga abortita, mentre tutti gli altri caratteri in arrivo verranno interpretati come comandi diretti, con le conseguenze che si possono immaginare.

Per ovviare a questa eventualità è comunque sufficiente impostare, nel programma di comunicazione che si adopera, una opzione **Expand Blanc** (in **JrComm** è disponibile), che trasforma una linea vuota in una serie di caratteri "spazio", evitando così l'arresto della ricezione.

Motivi tecnici a parte, non è poi da sottovalutare la negativa influenza sul collegamento dei disturbi psicocinetici prodotti dal sysop Red, da molti utenti erroneamente ritenuto umano, in realtà espulso dall'esplosione metafisica di un gigantesco foruncolo satanico. In questa prospettiva, ha ragione d'essere l'appellativo assegnatogli dal lettore: mitico, cos'altro...?

Posta "Principianti"

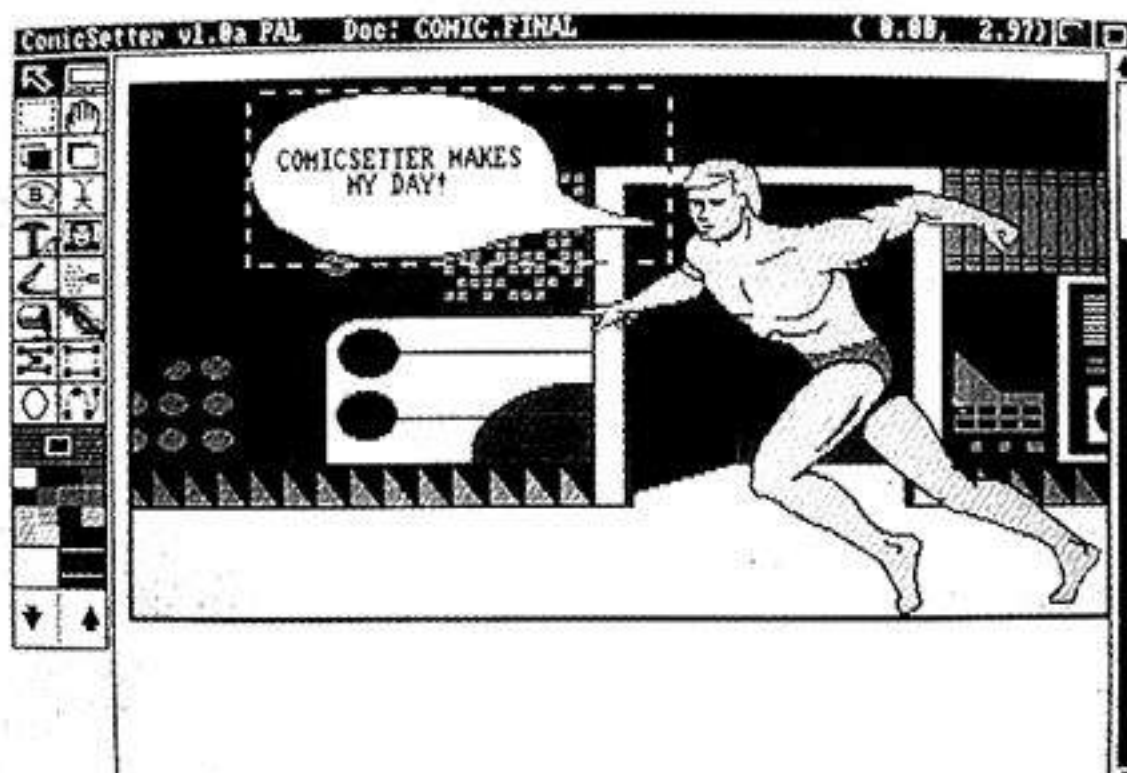
Un buon metodo può anche essere quello di apprendere i rudimenti sull'uso delle librerie dal manuale e da riviste come la nostra (che non lesina certo applicazioni pratiche sull'argomento), completando il tutto con l'acquisto del volumone **Libraries & Device** della Addison Wesley (un po' sproporzionato al basic, in verità) o di altre pubblicazioni riservate (ahimè) ad altri linguaggi. Da queste ultime possono, se non altro, essere ricavati tutti i dati sulla sintassi richiesta dalle varie funzioni di libreria.

Sarà dura in ogni caso, ma non c'è molta scelta...

Gia' fatto, o quasi

Vorrei ottenere, in basic, un effetto come quello adoperato per eliminare brani di testo dall'editor dello stesso AmigaBasic. E cioè adoperando il mouse, far diventare delle parti di schermo colorate in rosso, ma in modo che si veda il testo posizionato sotto il colore.
(Paolo Caruelli - Salerno)

Delimitare porzioni di schermo senza perderne il precedente contenuto, è un argomento già trattato su Postamiga del n. 80 (aprile 1990). In quella sede, si proponeva un metodo abbastanza semplice per ottenere riquadri "elastici" sullo stile delle finestre ridimensionabili di Intuition. La stessa tecnica allora affrontata, e sulla quale non ci soffermeremo più di tanto (si richieda eventualmente il fascicolo al nostro servizio arretrati), può adattarsi anche al tipo di esigenza proposta nella lettera. In pratica, l'unica differenza consisterebbe nell'ottenere riquadri colorati (o pieni che dir si voglia) piuttosto che "trasparenti".

Il che significa far ricorso alla **Graphic.Library**, o meglio alla sua funzione **SetDrMd** (che sta per Set Draw Mode), ampiamente descritta sul n. 80, per impostare un tracciamento **Jam2**. Questo, in pratica effettuerà un **Or escusivo** dei pixel interessati, mantenendo così il contenuto precedente dello schermo. Facendo seguire poi un comando basic come **Line** associato all'opzione di riempimento **bf** (si veda il manuale alla voce Line), il gioco è fatto.

Loquacita' Amiga

**** Non ho capito come si fa ad adoperare l'utility Say del Workbench per far parlare Amiga con voce femminile.***
(Dario Balmas - Pinerolo)

**** Siamo due lettori alle prime armi, vorremmo sapere se si può far parlare Amiga in Italiano.***
(Niki & Gianni)

Il programma **Say**, memorizzato nella directory **Utilities** del disco Workbench, se attivato da **Shell** senza alcun parametro (= digitando solo Say e premendo poi il Return) o se "biclickato" nella sua icona da ambiente Intuition (da Workbench, per intenderci), entra per così dire in modo interattivo, aprendo **due** finestre: una per l'input ed una che mostra l'output.

In quest'ultima, al momento del lancio, sono espresse piuttosto chiaramente (anche se in inglese) le opzioni da associare a Say per ottenere un mutamento del timbro della voce.

In pratica si tratta di inserire, prima della frase che si intende far pronunciare al computer, un trattino (il simbolo "**-**") seguito da una lettera ed eventualmente da un valore nel caso delle opzioni **-s** e **-p**. Per attivare la voce femminile, basterà p. es. digitare nella finestra di input o subito dopo il comando Say (da Shell): "**-f porca miseria**" (senza le virgolette). Si potrà constatare come il risultato non sarà molto soddisfacente, tanto nella timbrica che nella dizione, notoriamente basata sui **fonemi inglesi**.

Sul tipo di voce, si può anche agire tramite le opzioni "pitch" (**-p**) e "speed" (**-s**), per esempio adoperando lo stesso comando, prima visto, nel modo seguente:

```
Say -f -p250 -s130 porca mi-
seria
```

Lo si provi, variando gradualmente i valori associati a **-p** e **-s** fino ad ottenere il tono preferito, tenendo presente che **-s** ammette un "range" compreso tra 40 e 400, e **-p** tra 65 e 320.

Quanto alla possibilità di una **pronuncia italiana**, al momento non esiste un mezzo realmente valido per implementarla. Ci si può arrangiare, a patto di conoscere la pronuncia inglese dell'alfabeto, modificando l'input di conseguenza.

Si provi, ad esempio, a modificare l'input prima visto da "porca miseria" a "**porca mesareeea**": si avrà un certo avvicinamento al nostro idioma, ma nulla di più. In alternativa, ma anche in questo caso con risultati al momento non decisivi, è possibile adoperare una **Translator.library** adattata ai fonemi italiani. Per farlo, occorre anzitutto procurarsi questa versione di pubblico dominio della libreria, (per lo più rintracciabile con nomi tipo "**italian translator**"), e copiare il nuovo file Translator.library al posto di quello presente nalla directory **Libs** del disco Workbench. Dopo questa manovra, si può adoperare senza alcun problema tanto il comando **Say** che l'istruzione basic **Translate$**.

Se proprio si ha la necessità di inserire qualche frase in perfetto italiano in un proprio programma, non c'è però che un modo: armarsi di microfono e digitalizzatore audio.

Posta "Principianti"

Considerando però che si rende in ogni caso necessario l'uso della libreria grafica, si può adoperare, come alternativa a Line, la funzione **RectFill**, come mostrato nell'esempio pubblicato in queste pagine.

Il programma si limita a riempire la finestra di output con dei trattini, restando poi in attesa che si delimiti un riquadro col mouse. Questo sarà colorato al suo interno, ma senza alterare il contenuto sottostante.

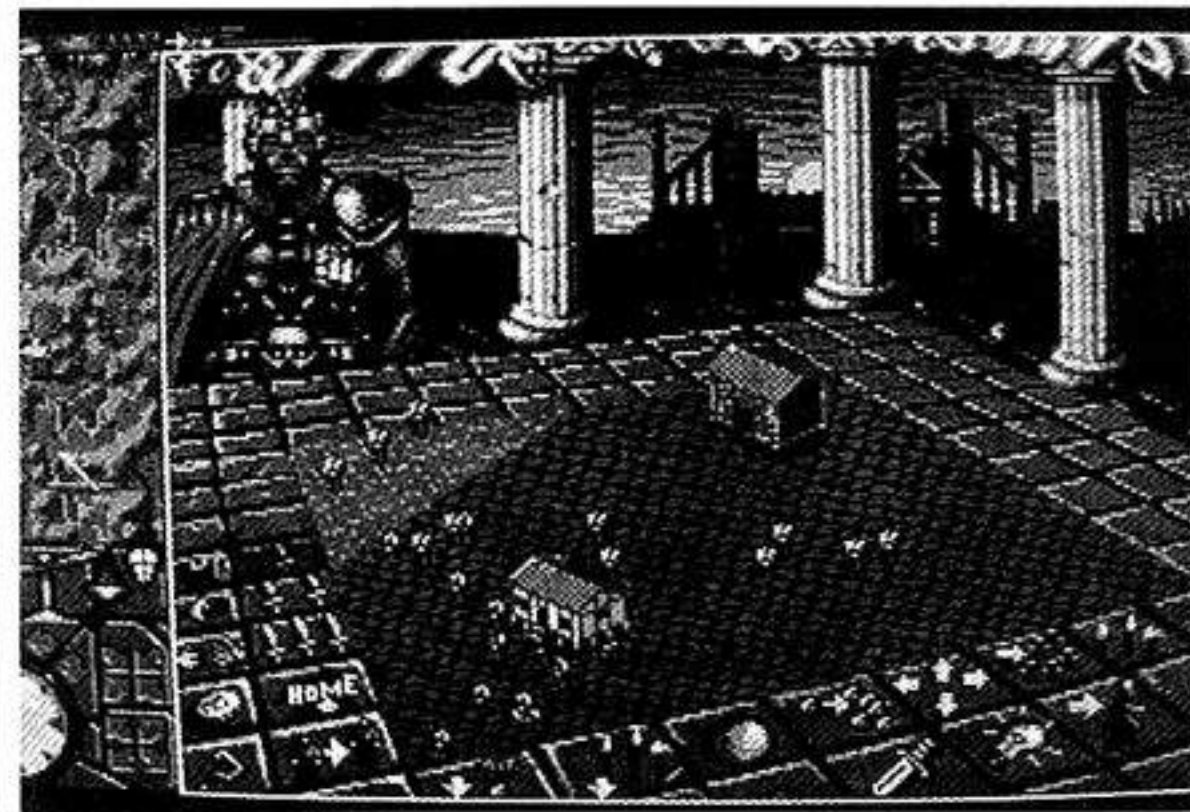
La funzione **RectFill&** richiede, come parametri, l'indirizzo della struttura **Raster Port** della finestra corrente (in basic: **window(8)**), e le coordinate dell'angolo superiore sinistro ed inferiore destro del riquadro da "riempire".

Chi è ancora alle prime armi non dimentichi che il listato dimostrativo (così com'è) richiede che nella directory corrente del basic sia presente il file **Graphics.bmap**, prelevabile dalla directory BasicDemos del disco Extras.

Funziona, eccome!

Ho acquistato il drive Cumana da 5.25 per Amiga da voi recensito sul n. 81, ma mi crea un sacco di problemi; spessissimo non riesce a leggere i files in formato Amiga...
(da una telefonata)

Un esemplare di quel modello di drive, prima di essere recensito, è rimasto costantemente "incollato" ad un Amiga per quasi un mese, assolvendo al proprio dovere senza mai presentare problemi. Rendendosi, tra l'altro, molto utile in più di una circostanza, tanto da far rimpiangere a chi scrive il momento in cui non ha potuto più disporne (sigh, non c'è più generosità in questo mondo crudele!).

Leggendo attentamente la recensione, si può forse risalire alla più probabile causa di anomalie: la **qualità dei flop-**

File misterioso

Ho notato che in molti dischi da me comprati, ma non in tutti, è presente un file di nome .fastdir. Non sarà per caso un virus? E se no, che cos'è, e a cosa serve?
(Tino Buscemi - Catania)

Non si tratta affatto di un virus, e l'unico relativo problema che gli si può imputare è l'occupazione di qualche (forse prezioso) kbyte nel dischetto. Talvolta, infatti, si può trovare uno di questi file anche in diverse subdirectory, se non addirittura in tutte. La presenza di **.fastdir** è legata all'uso di una delle più vecchie (seppur valide) "dir utility", di nome **Climate**, che consente una comoda gestione di files e floppy a suon di mouse: copia selettiva, installazione, lettura di file di testo o di immagini grafiche, eccetera.

Per evitare le lungaggini di AmigaDos, questo programma memorizza la lista dei files presenti nella directory cui si accede, in un file di nome appunto .fastdir, cosicché risulta poi molto più veloce una lettura del suo contenuto, piuttosto che la scansione fisica del disco. Se non si adopera Climate, si può tranquillamente eliminare il file con il comando **Delete** da Shell, o utilizzando... un'altra Dir Utility. In ambiente Pubblico Dominio esiste addirittura un vecchio programmino di nome **DirClear** (o similare), che provvede ad eliminare i files *.fastdir* da tutte le directory e subdirectory di un floppy, sempre che ne accerti la presenza.

Per la cronaca, c'è da dire che lo stesso Climate prevede, dal suo interno, l'inibizione all'uso di ".fastdir", che in questo caso non viene creato (né aggiornato, se esisteva già).

Posta "Principianti"

py. Questi, infatti, devono rigorosamente essere di **doppia densità e doppia facciata**. Detto in altre parole, si prestano a malfunzionamenti quei floppy da 5.25 che vengono venduti quasi... a peso, per un costo di solito inferiore alle 1000 lire. Con questo non si vuol dire che necessitano obbligatoriamente floppy di altissima qualità (da 10 mila lire cadauno, per intenderci), sarà sufficiente affidarsi a marche di larga diffusione, e spendendo giusto qualcosina in più. In fondo, sempre meno che per un (buon) dischetto da 3.5 pollici.

Voglio un Input!

Anche se seguo tutti i vostri articoli riguardanti AmigaDos, non ho ancora capito se è possibile fare accettare un input da un batch file. Non parlo del banale "Y" e "N" del comando Ask né dei parametri che si possono immettere dopo Execute, ma di una vera e propria stringa o valore numerico come l'Input del basic.
(G. Rutigliano - Torino)
(Numerose altre lettere)

Un input generico, come più volte mostrato in precedenti numeri della rivista, può anche essere introdotto facendo seguire un punto interogativo ad un comando del Dos. In questo caso, però, l'input avrà per così dire una rilevanza "locale", ovvero riferita solo all'applicazione di quel comando. In pratica, se p. es. si inserisce nell'ambito di un batch file qualcosa come **CD >nil: ?**, l'esecuzione si arresterà in attesa della digitazione (seguita da un Return) del parametro da associare al comando CD, che poi verrà attivato come di norma.

Per qualcosa di più complesso, paragonabile alla "basica" acquisizione di una variabile durante lo svolgimento di un programma, è necessario fondere questa tecnica all'uso delle variabili ambiente del Dos. Argomento, quest'ultimo, trattato proprio sul numero scorso di CCC, nella sezione di AmigaFacile dedicata al device **Env:**.

Rimandando all'esame di quelle pagine per una più dettagliata descrizione del device e delle variabili, basterà qui ricordare come queste possono essere assunte adoperando il comando **Setenv**. Si provi dunque, dopo avere aperto una finestra Shell dal disco Workbench, ad impartire...

```
Setenv ?
```

Apparirà la stringa del cosidetto "template", e si potranno immettere, nell'ordine, il nome della variabile e la stringa da associare ad essa.

Per ottenere in un batch file l'input del solo assegnamento ad una variabile interna, si renderà però necessario:

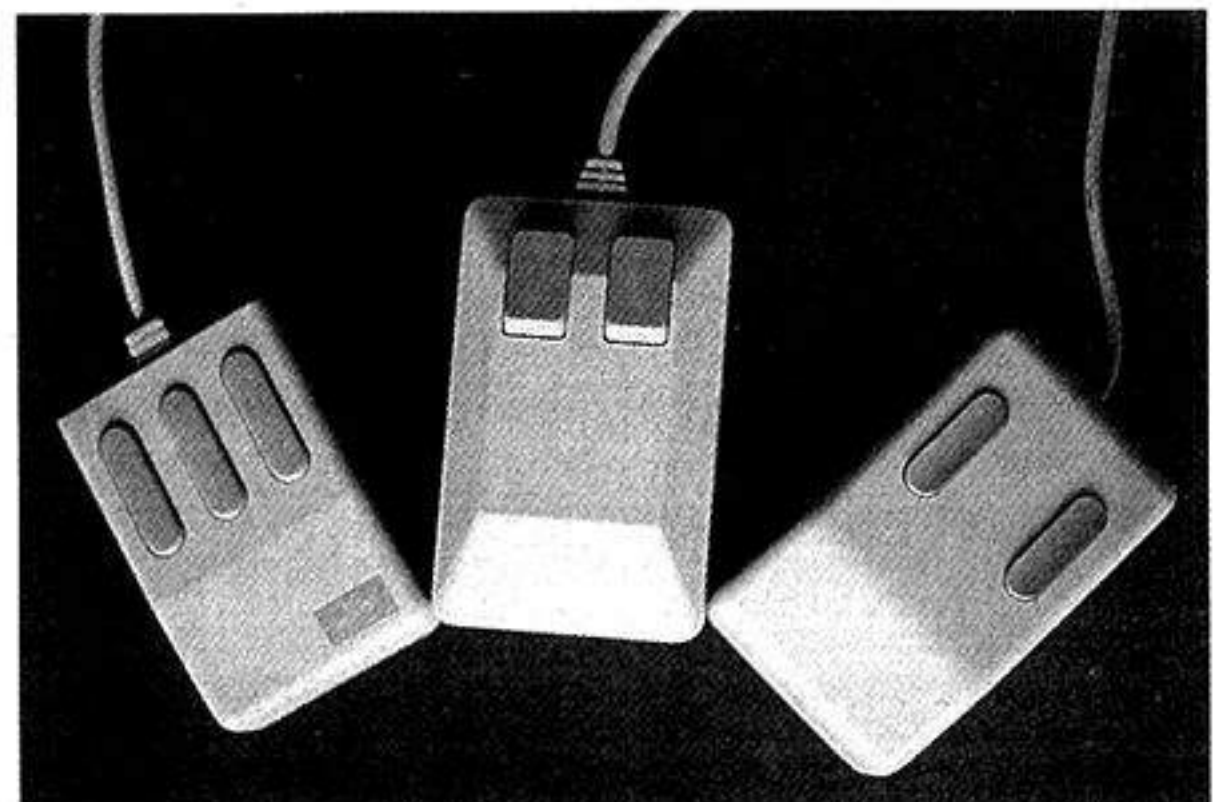
1) abolire la comparsa della stringa.

2) far sì che il primo parametro di Setenv venga acquisito dall'interno, ovvero che non debba essere digitato dall'utente.

Il primo punto è di semplice attuazione, in quanto basterà redirigere verso **Nil:** l'output del comando. Per il secondo, sarà necessario ricorrere alla precisazione del template. Ovvero (si veda sempre AmigaFacile del numero scorso), visto che i due parametri (come mostrato dopo SetEnv ?) sono precisati dalla stringa **Name/a,String**, andrà adoperata una forma...

```
Setenv >nil: Name=X ?
```

...con **X** che rappresenta il nome della variabile. Quanto si immetterà da tastiera dopo un comando simile, diventerà in pratica il contenuto della variabile. Per non restare sul va-

go, ecco per esempio un banale file batch che consente una scelta da menu:

```
echo "A - Pippo"
echo "B - Pluto"
echo "C - Minnie"
echo " "
echo "Scegli"
setenv >nil: name=X ?
if $x eq "A"
echo "hai scelto Pippo"
endif
if $x eq "B"
echo "hai scelto Pluto"
endif
if $x eq "C"
echo "hai scelto Minnie"
endif
```

In questa applicazione, la variabile X conterrà la lettera digitata (**a**, **b**, oppure **c**), che potrà essere vagliata da **If** per attivare solo una certa sezione del batch. In definitiva, un input del tutto analogo a quanto implementabile in veri linguaggi di programmazione.

Quanti Assign?

Se voglio far cercare dei comandi Dos in altre directory oltre la C, adopero Path e tutto è risolto. Ma si può fare qualcosa del genere con Assign? Se, per esempio, creo una directory Libs in Ram disk e impartisco Assign Libs: Ram:libs, le librerie verranno cercate nella nuova directory in ram, ma non più in quella del disco. Il mio problema è che, avendo poco spazio nel disco di boot, non vi posso inserire (p. es.) la Arp.library. Non si può fare in modo che il sistema cerchi le librerie in più directory, così potrei metterne alcune in un disco diverso da quello di boot, senza doverle ricopiare tutte?
(A. Scandurra - Milano)

In effetti una feature come quella auspicata dal lettore farebbe comodo in più di una occasione, ma purtroppo Assign non la consente. L'unica soluzione, ricorrendo ai comandi del dos, consiste proprio nel creare e copiare altrove la directory **Libs** di sistema (in Ram disk o in un altro floppy), aggiungervi eventuali altre librerie, ed impartire un comando...

`Assign Libs: nome:libs`

...con **nome** che identifica il nome del disco o dell'unità prescelta.

Rivolgendosi, però, al solito toccasana rappresentato dal software di Pubblico Dominio, vi si può rintracciare uno speciale comando di nome **AsPath**, che è proprio un Assign in grado di creare assegnamenti **multipli**, o di aggiungere nuovi "percorsi" (proprio come Path) a quelli preesistenti. Il file è rintracciabile soprattutto tramite le molte banche dati (bbs) italiane che "supportano" il pubblico dominio, per lo più con nome **ASSxx.lzh** (xx specifica la versione), ed è fornito di una sua documentazione interna in **italiano**, in quanto opera del nostro connazionale **Romano Tenca** e della sua soft house **Anarkick**. Caldamente consigliato per chi sfrutta il Dos... fino all'osso, può anche essere prelevato direttamente dalla nostra BBSystems.

Shell a tutto schermo

Come agire per avere la finestra Shell che occupa l'intero schermo subito dopo aver clickato nella sua icona, o comunque senza doverla allargare col mose?
(firma illeggibile)

Se si prevede un accesso a Shell attraverso la sua icona nel disco Workbench, basterà seguire questa procedura:

* Clickare una sola volta nell'icona.
* Selezionare "Info" dal menu Workbench.
* Clickare su "Add" nel riquadro Tool Types della finestra Info.
* Digitare all'interno del riquadro la seguente istruzione su un unico rigo e adoperando il maiuscolo e gli spazi così come sono:

`WINDOW=newcon:0/0/640/255/MyWork`

* Clickare sull'opzione **Save**.

Tutto fatto. Come ovvio, **MyWork** è un nome fittizio che può essere liberamente modificato. E' consentito inserire eventuali spazi nel nome, senza altre modifiche alla stringa comando. Il settaggio appena visto è da riferirsi ad uno schermo in alta risoluzione non interlacciata, ovvero di "**80** colonne". Qualora si adottasse la bassa risoluzione (**60** colonne), il **640** della stringa di comando andrebbe sostituito con **320**.

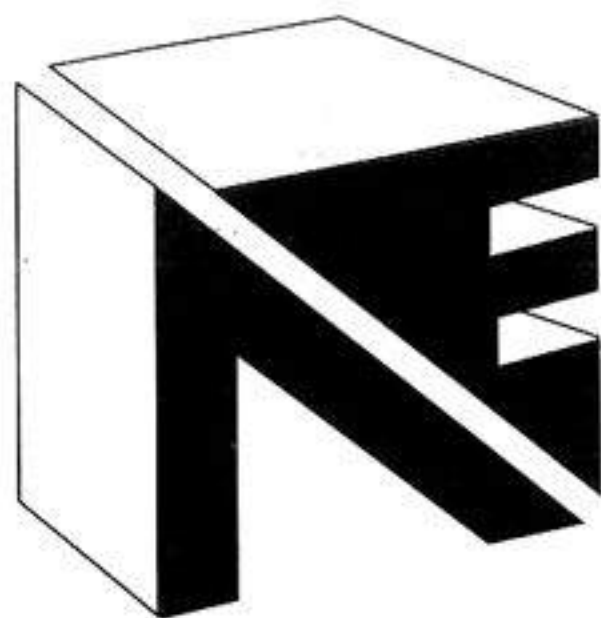
Per ottenere lo stesso effetto, ma attivando eventuali nuove finestre da ambiente Dos, si renderebbe invece necessario impartire ogni volta un prolisso comando...

`Newshell Newcon: 0/0/640/255/Nome`

Per evitarlo, risulta più comodo creare un **Alias** una volta per tutte, ovvero un comando...

`Alias Newshell Newshell newcon: 0/0/640/255/Nome`

In tal modo, ogni volta che si invocherà Newshell, la nuova finestra verrà aperta a tutto schermo. Per avere sempre disponibile questa feature, la soluzione migliore consiste nell'inserire il comando appena visto nel batch file **Shell-Startup** presente nella directory **S** del disco Workbench.

A differenza di quanto detto a proposito della stringa di comando inserita nell'info dell'icona, l'uso di eventuali spazi nel nome della finestra obbliga all'uso dei doppi apici (virgolette) per racchiudere tutto il parametro del comando Newshell, come spiegato ampiamente negli ultimi appuntamenti con Postamiga e AmigaFacile.

NEWEL® srl

computers ed accessori

20155 MILANO via Mac Mahon, 75

NEGOZIO tel.02/323492
UFFICI tel.02/3270226
FAX 24h tel.02/33000035
UFFICIO SPEDIZIONI tel.02/33000036

UNICA SEDE IN ITALIA

VENDITA ANCHER PER CORRISPONDENZA IN TUTTA ITALIA

EVASIONE ORDINI NELLE 24 ORE SUCCESSIVE ALL'ORDINE

ACCESSORI PER COMPUTER

- SUPPORTO MONITOR MASCULANTE 14" **L.39.000**
- SOTTOSTAMPANTE UNIVERSALE 80 COLONNE **L.25.000**

JOYSTICK IN OFFERTA

SUPER JOYSTICK
Dagli USA, il robustissimo "Super Joystick anima in metallo, sei microswich di precisione, bottone di fire sull'impugnatura e sulla base, autofire disinseribile tramite swich, il tutto in un nuovissimo design nero con ventose.
L. 35.000

JOYSTICK "CLOCHE" per simulatori di volo
Altra novità dagli Stati Uniti questo [illegible] programmi tipo: Flight Simulator, Mig 29, Falcon, F16 Combat Pilot, F19 Stealth Fighter e molti altri
L. 69.000

MOUSE SELECTOR

Utilissimo permette di collegare contemporaneamente il mouse ed il joystick e selezionare tramite interuttore quello desiderato, senza dover ogni volta sconnettere e rischiare di danneggiare il computer. **L. 29.000**

HARDCARD 2091 (SOLO CONTROLLER) L. 330.000
HARDCARD 2091 + 45 MB HARDISK "QUICKLOAD" L. 875.000

Hard-disk Commodore nuovo veloce controller, espandibile a 2 MB, il tutto su CARD, semplicissima installazione, autoboot, il tutto ad un prezzo eccezionale.

DIGITALIZZATORI VIDEO

DIGI VIEW 4,0 GOLD
Digitalizzatore professionale il più affidabile per risultati incredibili, a colori in altissima risoluzione, filtratura manuale, completo di istruzioni e software originale **L. 290.000**

VIDEON 3,0
Nuovo digitalizzatore professionale a colori ara a 32000 colori, con SUPER-VHS, nuovo software di gestione per creare sempre immagini più incantevoli lavoro con immagine ferma per almeno 13 secondi, completo di cavi e istruzioni in italiano **IN OFFERTA**

CENTINAIA DI PROGRAMMI, SEMPRE ULTIME NOVITÀ "IMPORTAZIONE DIRETTA" DAGLI USA, INGHILTERRA, GERMANIA, "GIOCHI, UTILITY, SOFTWARE MIDI".

NEWEL PRESENTA UN NUOVO SERVIZIO, TUTTO IL MEGLIO DEL SOFTWARE DIRETTAMENTE A CASA TUA, LA PIU' VASTA GAMMA DI SOFTWARE ORIGINALE!!! SOFTWARE INTERAMENTE IN ITALIANO:

- **DELUXE PAINT III** Programma grafico pittorico L. 149.000
- **DELUXE VIDEO III** Video, animazione, Videoclip L. 169.000
- **PAINTER 3D CAD 3D**, per arredamento ecc. L. 149.000
- **SUPERPLAN (SUPER LOGISTIX)** Tipo lotus 123, foglio elett. L. 189.000
- **SUPERBASE PERSONAL** Potente archivio multiuso L. 179.000
- **SUPERBASE PROFESSIONAL** Database, prof. Programmabile L. 299.000
- **C1-TEXT 3.0** Il più potente Wordprocessor L. 89.000
- **PERSONAL FONT MAKER** Crea fonts/caratteri come vuoi L. 99.000
- **PHOTON PAINT II** Programma grafico pittorico L. 89.000
- **TOTO MANIA** Elaborazioni sistem Totocalcio L. 59.000
- **PROFESSIONAL PAGE 1.3** Il più potente DeskTop Pubblish. L. 490.000
- **AMIGA "APPETIZER"** Videoscrittura + Disegno + Music L. 39.000

Tutti i sopraelencati programmi sono interamente in italiano sia il programma che le istruzioni.

Confezione 200 etichette per floppy disk, colorate appositamente studiate per dischi 3 1/2 con un speciale collante che non danneggia i dischetti. L.19.000

INTERFACCIA 4 JOYSTICK

Permette di collegare contemporaneamente 4 joystick all'amiga e quindi di giocare in 4 contemporaneamente a giochi tipo CALCIO, PALLAVOLO ecc.
DISPONIBILE !!! **L. 29.000**

SUPER 64 EMULATOR

Ultima versione del famoso emulatore C64, completo di interfaccia hardware per la connessione con le periferiche del 64
L. 29.000

AMIGA PENNA OTTICA (OFFERTA DEL MESE!)

Divertente, permette di usare molti programmi grafici, come deluxe paint ecc. Disegnando direttamente sul video, molto semplice da usare, istruzioni in italiano.
L. 29.000

ACCESSORI VARI COMMODORE

A501	ESPANSIONE COMMODORE	L.	130.000
A520	MODULATORE TV-PAL	L.	49.000
A2088	JANUS XT + DRIVE + DOS 4.1	L.	590.000
A2286	JANUS AT + DRIVE	L.	1.390.000
A2010	DRIVE PER A2000 COMMODORE	L.	195.000
A2060	SCHEDA MODULATORE TV A2000	L.	179.000

ACTION REPLAY 2 (disponibile anche per AMIGA 2000)

la prima cartuccia rivoluzionaria multifunzioni per amiga 500/1000 con opzioni di freeze: permette di sproteggere la maggior parte dei programmi in commercio (consentendoti di creare giochi di sicurezza per uso personale, inoltre permette di creare giochi trainer, (vite infinite ecc.), permette di bloccare un gioco in qualsiasi momento dal medesimo posto, salva una qualsiasi videata (disegno, testo) su disco, consentendoti una facile hardcopy anche su stampante, funzione moviola (rallenta programmi e giochi), potente virus-detector, sprinteeditor, oltre che ad un monitor straordinario per il linguaggio macchina, questo e molto, molto di più, ti aspetta in amiga action replay!!! il tutto ad un prezzo eccezionale! versione originale con manuale in italiano **L. 169.000**

TURBO ACCELERATOR CARD PER AMIGA 500/1000/2000

Funziona su tutti gli amiga, semplice installazione, non necessita di saldature, 100% compatibile, velocità selezionabile tra 7 e 14 Mhz, 32 kb di ram statica ad alta velocità, di cui 16 come cache memory. Il più veloce acceleratore senza memoria a 32 bit. STRAORDINARIO!!! **L. 450.000**

AMIGAVISION (NOVITÀ)

Finalmente disponibile anche in Italia il straordinario Pacchetto che fino ad oggi corredava soltanto l'Amiga 3000, questo eccezionale programma per la programmazione multimediale, grafica, animazione, video, musica e molto più. Regala al tuo Amiga questo fantastico programma!!!
L. 149.000

BOOTSELECTOR

Trasorma il drive esterno in DFO: (interno) utile per evitare l'usura eccessiva del drive interno, e risolvendo inoltre problemi di compatibilità con il drive originale, Kit di semplicissima installazione. **L. 19.000**

MOUSE PAD - TAPPETINO ANTISTATICO PER MOUSE

Utilissimo tappetino antistatico per mouse, indispensabile per un buon utilizzo del mouse, evita il contatto diretto con superfici non idonee per un corretto uso del mouse. **L. 15.000**

CONTROLLER GVP SERIE II PER A2000

Il nuovissimo controller della Great Valley Production per Harddisk con le nuove FASTROM. La scheda è espandibile con moduli Simm fino ad 8 Mb di ram, installazione semplicissima ed velocissima. **L. 490.000**
OGNI MODULO DI RAM DA 2MB L. 275.000

A-2320 FLICKER FIXER

Elimina il fastidissimo sfarfallio del Amiga in altissima risoluzione, indispensabile per chi lavora in grafica, è indispensabile un monitor multisync o VGA. **L.430.000**

FISH-DISK AGGIORNATI AL N° 400 TUTTI! IN OFFERTA

ALLA NEWEL TROVI TUTTI I VIDEOGIOCHI RECENSITI SU QUESTA RIVISTA, IL PIU' GRANDE ASSORTIMENTO DI SOFTWARE ORIGINALE PER IL TUO COMPUTER!!!

OFFERTA: ACQUISTANDO 2 PROGRAMMI A SCELTA SI HA DIRITTO AD UNO SCONTO DEL 10% SU ENTRAMBI I PROGRAMMI!!!

di Domenico Pavone

Rad; ce n'e' per tutti!

Un file batch per installare in completo automatismo, e con qualunque configurazione di memoria, una ram disk resistente al reset

A giudicare dal gran numero di richieste pervenute in redazione, la **Recoverable Ram** (per gli amici: RAD) suscita, tra gli utenti più smaliziati, un interesse pari solo alla rapidità del suo... abbandono.

Il motivo del declino, nella maggior parte dei casi, è legato soprattutto alla configurazione di memoria posseduta: non tutti possono permettersi svariati megabyte di espansione, ed è opinione corrente che un Amiga in versione base non possa sfruttare appieno le potenzialità di questa periferica logica. Una prima anticipazione: non è così, o almeno non del tutto.

Per altri, invece, i problemi cominciano a manifestarsi quando si tratta di passare alla pratica. Di memoria, magari, se ne dispone a iosa, ma sfruttare concretamente una speciale ram disk, soprattutto in funzione della sua capacità di autoboot (con sistema operativo **1.3**), è un altro discorso: significa andare a rovistare nella mountlist, nelle varie directory indispensabili per il funzionamento di Amiga, ed altre amenità simili. Niente di trascendentale, ma per chi è alle prime armi la cosa può presentare parecchi trabocchetti.

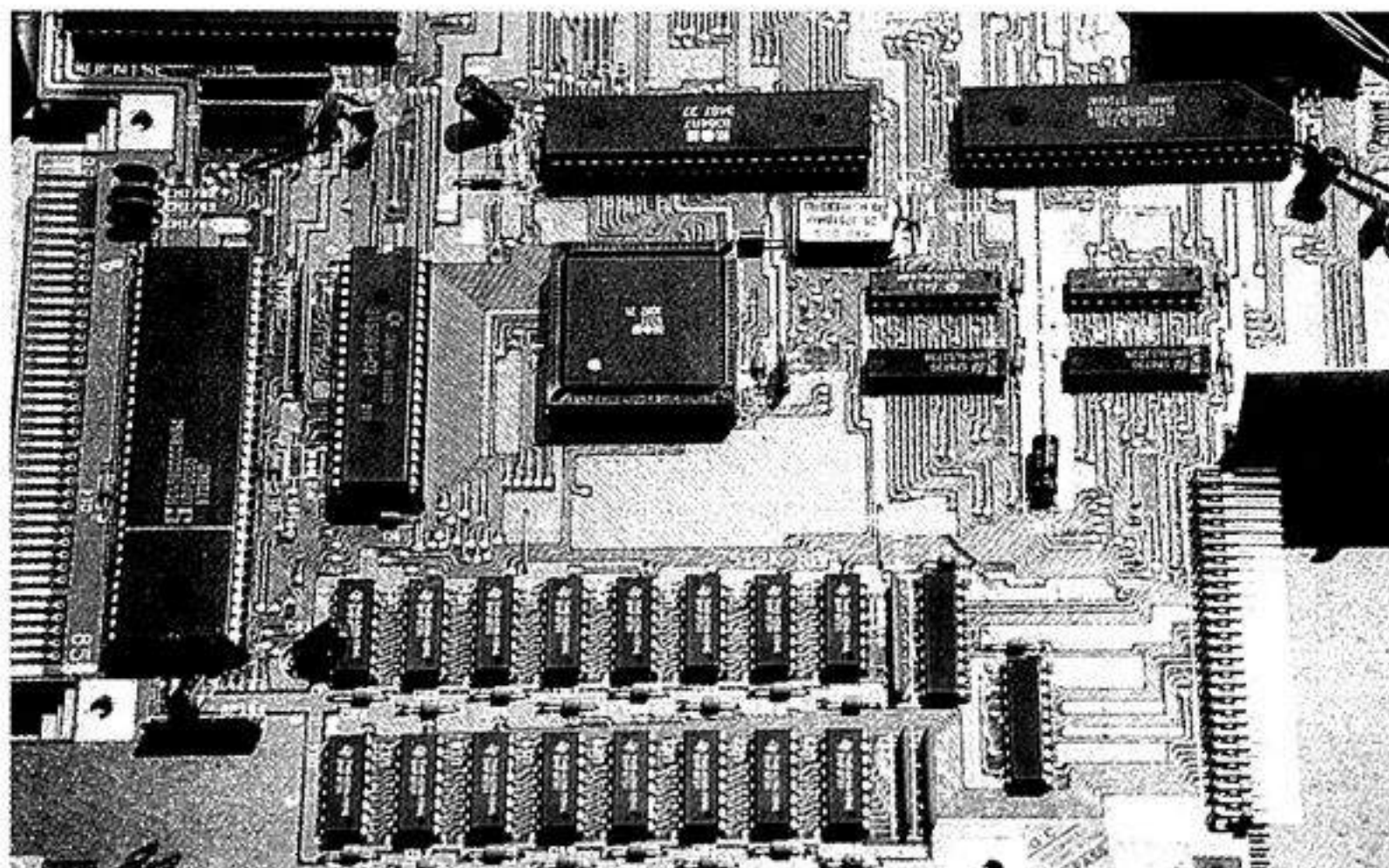
Le implicazioni teorico-pratiche riguardanti la Rad sono in effetti già state trattate nell'ormai lontano **N. 66** della rivista (se ne consiglia in ogni caso una "rivisitazione"), che illustrava una serie di interventi finalizzati alla implementazione di una Ram Disk resistente al reset, ed anche bootabile. In questa sede, invece, sarà preponderante l'aspetto pratico, proponendo un unico file batch in grado di soddisfare le esigenze di chiunque, per di più con estrema semplicità. Senza altri interventi di sorta, sarà cioè sufficiente copiare il listato di queste pagine, mandarlo in esecuzione, e scegliere da menu il tipo di Rad desiderata. Nient'altro.

Prima di entrare nei dettagli, va detto che le dimensioni del file non sono proprio irrisorie, per cui si consiglia una certa attenzione nel copiarlo: anche piccoli errori come l'assenza di un simbolo "due punti" possono inficiarne il funzionamento!

Uso e consumo

Prima di esaminare a fondo come funziona il file, con tutte le implicazioni "dossiane" del caso, vediamo di soddisfare i meno propensi alla teoria, illustrando che cosa fa il batch in questione, e, soprattutto, come farglielo fare.

Intanto, come già detto, si tratta di un file batch, ovvero di una serie di comandi del Dos raggruppati a formare una sorta di "programma". Il che significa che andrà editato in ASCII, cioè copiato adoperando l'**Ed** del Dos (si veda Amigafacile sul **n. 78**), oppure un qualunque altro editor o word processor che produca un file in ascii puro (per esempio. C1-Text, Cygnus, Az, ecc.).

Come i lettori che ci seguono da più tempo ormai sapranno, il batch può essere memorizzato ovunque, ma è preferibile sfruttare la **directory S** perché sempre raggiungibile dal comando **Exe-**

```
;          FILE BATCH 'MULTIRAD'
;     Attivazione:  Execute Multirad
; Non richiede nessun altro intervento!
;---------------------------------------------
copy c:copy ram:
cd ram:
copy c:echo ram:
echo "*ec Attendi..."
copy c:join ram:
copy c:if ram:
copy c:endif ram:
copy c:delete ram:
copy c:assign ram:
copy c:makedir ram:
copy c:skip ram:
assign env: ram: ; nel caso mancasse...
;---------------------------------------------
lab start
echo "*ec "
echo "A - Rad di 264K con tutti i comandi"
echo "    del Dos velocizzati oppure vuota."
echo "    Non bootabile."
echo "    Indicata con soli 512K di ram.*N"
echo "B - Rad di 330K con tutti i comandi"
echo "    Dos e 80K (di rad) liberi, oppure"
echo "    vuota. Non bootabile."
echo "    Indicata con 1 Mb di ram.*N"
echo "C - Rad di 330K con sistema bootabile"
echo "    e 29 K (di rad) liberi."
echo "    Indicata con 1 Mb di ram.*N"
echo "D - Intero sistema in Rad di 880K."
echo "    Indicata con almeno 1.5 Mb di ram."
echo "    Bootabile. Creazione molto rapida.*N"
echo "E - Intero sistema in Rad di 880K con"
echo "    altri 880K di Rad liberi. Bootabile."
echo "    Indicata con piu' di 2 Mb di ram.*N"
echo "*e[33mDigita la scelta e premi Return"
;---------------------------------------------
setenv >nil: name=x ?
;---------------------------------------------
if $x eq "A"
echo >8 "     LowCyl = 0 ; HighCyl = 23"
skip cont
endif
if $x eq "B"
echo >8 "     LowCyl = 0 ; HighCyl = 29"
skip cont
endif
if $x eq "C"
echo >8 "     LowCyl = 0 ; HighCyl = 29"
skip cont
endif
if $x eq "D"
echo >8 "     LowCyl = 0 ; HighCyl = 79"
skip cont
endif
if $x eq "E"
```

cute, necessario per mandarlo in esecuzione. Schematizzando, si potrà in definitiva:

1) salvarlo in una directory o disco qualsiasi. In questo caso andrà lanciato citandone per esteso il percorso, a meno che non sia la directory corrente. Supponendo si sia assegnato il nome **Multirad** al file, andrà per esempio adoperata una forma del tipo...

`Execute df1:Multirad`

...se il file è nel disco inserito nel secondo drive.

2) salvarlo nella directory **S** del disco adoperato per il boot, che (soprattutto per i non esperti) dovrà *necessariamente* essere il **Workbench 1.3**. Per attivarlo, basterà un banale `Execute Multirad`.

3) salvarlo nella directory **C** del Workbench. Per sfruttare questa possibilità, è indispensabile poi adoperare da shell un comando...

`Protect c:Multirad +S`

...che renderà autoeseguibile lo script. In altre parole, potrà poi essere attivato digitando solo `Multirad`, quale che sia la directory corrente.

Il file richiede una unica condizione: che si adoperi una copia del disco Workbench 1.3 per bootare Amiga. Per proprie esigenze, questo può anche essere parzialmente rimaneggiato, ma se non si è in grado di capire a fondo come funziona il batch, è opportuno lasciarlo integro, o giusto eliminare (per esempio) il **Notepad** per fare un po' di spazio.

Prima di attivare Multirad, è inoltre consigliabile allargare la finestra Shell nella quale si opera fino alle sue dimensioni massime, per consentire al menu iniziale del batch di apparire integralmente. Questo è infatti costituito da **5 voci**, la cui selezione è affidata alla digitazione del corrispondente carattere alfabetico seguito da Return. Se, per errore, si digita una scelta non compresa nel menu, quest'ultimo viene ripresentato senza altre conseguenze, mentre per interrompere la prosecuzione del file sarà sufficiente premere **Ctrl** + **C** (e return).

Come chiaramente espresso nel menu, le prime due scelte installano una Rad non bootabile, ma che può risultare ugualmente comoda. A seconda della risposta ad una successiva richiesta di

```
echo >8 "      LowCyl = 0 ; HighCyl = 159"
skip cont
endif
skip start back ; se errore di scelta
;-------------------------------------------
lab cont
echo "*ec Attendi...."
echo >1 "RAD: Device = ramdrive.device"
echo >2 "     Unit    = 0"
echo >3 "     Flags   = 0"
echo >4 "     Surfaces  = 2"
echo >5 "     BlocksPerTrack = 11"
echo >6 "     Reserved = 2"
join 1 2 3 4 5 6 to mnt1
echo >7 "     Interleave = 0"
echo >9 "     Buffers = 20"
echo >10 "     BufMemType = 1"
echo >11 "     Mount = 1"
echo >12 "#"
;-------------------------------------------
join 7 8 9 10 11 12 to mnt2
join mnt1 mnt2 to mountrad
;-------------------------------------------
mount rad: from mountrad
;-------------------------------------------
if $x eq "A"
skip make1
endif
if $x eq "B"
skip make1
endif
if $x eq "C"
skip make2
endif
if $x eq "D"
skip make3
endif
if $x eq "E"
skip make4
endif
;-------------------------------------------
lab make1
Ask "*ec Preferisci una Rad vuota? (y/n)"
if warn
delete >nil: ram:#?
echo "*ec *e[33m"
info
echo "*N RAD INSTALLATA *N*e[0m"
cd sys:
quit
endif
;-------------------------------------------
echo "*ec Attendi...."
copy >nil: sys:c rad:c
assign c: rad:c
delete >nil: ram:#?
cd sys:
```

input, si può infatti optare per una Rad vuota, oppure contenente i comandi del Dos (per intenderci: quelli compresi nella directory C di sistema).

Nel primo caso, si potrà utilizzare il device virtuale (anche da Workbench) per memorizzarvi file che verranno letti molto più rapidamente che da disco. La velocità della Rad è inferiore a quella della normale Ram Disk, ma con l'enorme vantaggio che il reset (o una Guru di troppo) non ne distrugge il contenuto.

Scegliendo invece di mantenervi i comandi del dos, il loro accesso sarà estremamente rapido, consentendo soprattutto, a chi predilige l'ambiente Shell, di affrancarsi dai dischetti. Chiaro che, in questa ipotesi, lo spazio disponibile all'interno della Rad risulterà drasticamente ridotto, a seconda della scelta adottata. Per un **Amiga 500 non dotato di espansione**, sarà comunque obbligata la scelta **A**, mentre con 1 Megabyte di ram disponibile le cose possono farsi più ghiotte.

Con l'**opzione C**, infatti, si può disporre di un sistema rebootabile, veloce, e completamente svincolato dalla presenza dei dischetti. Chiaro che, dovendo "giostrare" solo 1 Mb, viene mantenuto in Rad solo l'indispensabile, ma ce n'è abbastanza per adoperare Amiga come di consueto, tanto da Workbench che da Shell. In questo caso, dopo avere selezionato l'opzione, basterà solo attendere la conclusione dei "lavori", togliere il disco dal drive **df0:**, e resettare il computer. Null'altro. Il tempo di esecuzione rientra nell'ordine dei **2 minuti**, ma il risultato finale non li farà rimpiangere di certo: ad ogni nuovo reset Amiga impiegherà un tempo di gran lunga inferiore per eseguire la procedura di start, per non parlare della nuova disponibilità dei drive, liberi per qualunque altra applicazione.

Le opzioni **D** ed **E**, sono invece riservate per chi dispone di molta più memoria ram, da **1,5 Mb** in su. In entrambi i casi si ottiene una Rad rebootabile, naturalmente molto più completa di quanto visto a proposito dell'opzione C per 1 Megabyte di Ram. In pratica, equivarrà all'intero contenuto del floppy Workbench.

Compatibilmente con la propria configurazione hardware, si consiglia di sperimentare più di una soluzione, che in ogni caso può essere variata volta per

```
echo "*ec*e[33m"
info
echo "*N TUTTI I COMANDI SONO IN RAD *N*e[0m"
quit
;-------------------------------------------
lab make2
makedir rad:s
ask "*ecHai la tastiera italiana? (y/n)"
if warn
echo >1 "Setmap i"
setenv tast I
else
echo >1 "setmap usa1"
endif
echo "*ec Attendi...."
echo >2 "assign env: ram:"
echo >3 "Setclock load"
echo >4 "resident CLI l:shell-seg System pure add"
echo >5 "resident c:execute pure"
echo >6 "mount newcon:"
echo >7 "ff -0"
echo >8 "loadwb"
echo >9 "endcli"
join 1 2 3 4 5 6 7 8 9 to rad:s/startup-sequence
;-------------------------------------------
makedir rad:L
copy L:disk-validator rad:L
copy L:Port-handler rad:L
copy L:Shell-seg rad:L
copy L:Newcon-handler rad:L
copy L:Ram-handler rad:L
makedir rad:devs
copy devs:mountlist rad:devs
copy devs:printer.device rad:devs
copy devs:parallel.device rad:devs
copy devs:serial.device rad:devs
copy devs:ramdrive.device rad:devs
copy devs:system-configuration rad:devs
makedir rad:devs/printers
makedir rad:devs/keymaps
if $tast eq "I"
copy devs:keymaps/i rad:devs/keymaps
else
copy devs:keymaps/usa1 rad:devs/keymaps
endif
copy >nil: devs:printers/#? rad:devs/printers
makedir rad:libs
copy libs:icon.library rad:libs
copy libs:info.library rad:libs
makedir rad:c
copy c:cd rad:c
copy c:assign rad:c
copy c:avail rad:c
copy c:copy rad:c
copy c:date rad:c
copy c:delete rad:c
copy c:dir rad:c
```

volta, a seconda dell'uso che si pensa di fare della Recoverable Ram.

Tutte le opzioni, come ovvio, si prestano a personalizzazioni di vario genere, ma anche lasciando invariato il batch qui proposto si può trarne notevoli vantaggi. Smanettare tra i vari comandi che lo compongono, d'altra parte, richiede che si comprenda almeno superficialmente come il file ottiene lo scopo. Vediamolo meglio, stavolta con qualche approfondimento più tecnico.

Una Rad addomesticata

Poco da dire sulla prima parte del batch file, che si occupa banalmente di copiare in Ram Disk i comandi che andranno adoperati con maggiore frequenza (tra cui **Echo**), in modo da velocizzare il suo svolgimento. Come ovvio, perché questi comandi vengano via via caricati dalla Ram, e non da disco, il batch provvede a settare come directory corrente proprio la Ram (adoperando il canonico **Cd**).

Stampato a video il menu, si occupa poi di prelevare l'input da tastiera, assegnandolo ad un variabile interna di nome **X**. Si tratta in pratica di adoperare opportunamente il comando **Setenv**, così come descritto nella rubrica **Postamiga** su questo stesso numero della rivista. Ciò fatto, si entra nel vivo dell'argomento Rad, che richiede una breve premessa.

Questa periferica virtuale, come dovrebbe essere noto, è una struttura "rigida", non elastica come la normale Ram disk. In quanto tale, è necessario predisporre la sua capienza prima di installarla. Quella che sarà la sua occupazione fisica di memoria (e quindi le sue dimensioni), come pure una serie di altre caratteristiche delle quali possiamo anche non occuparci, viene determinata dal contenuto di un file particolare, di nome **Mountlist**, presente nella directory **Devs** del disco Workbench. Quando si adopera il comando **Mount Rad:** per installare la Recoverable Ram, il sistema in pratica legge questo file e in base al suo contenuto organizza il device Rad.

Se si prova a digitare (da Shell)...

`Type Devs:Mountlist`

...si noterà come esiste nel suo contesto una "entry" di nome Rad, con una riga che precisa `LowCyl=0 ; HighCyl=21`. Ebbene, modificando il valore di **Hi-**

```
copy c:ed rad:c
copy c:else rad:c
copy c:endcli rad:c
copy c:endif rad:c
copy c:execute rad:c
copy c:ff rad:c
copy c:iconx rad:c
copy c:if rad:c
copy c:info rad:c
copy c:install rad:c
copy c:list rad:c
copy c:loadwb rad:c
copy c:makedir rad:c
copy c:mount rad:c
copy c:newcli rad:c
copy c:newshell rad:c
copy sys:system/nofastmem rad:c
copy c:path rad:c
copy c:prompt rad:c
copy c:protect rad:c
copy c:relabel rad:c
copy c:rename rad:c
copy c:resident rad:c
copy c:run rad:c
copy c:setclock rad:c
copy sys:system/setmap rad:c
copy c:stack rad:c
copy c:type rad:c
makedir rad:system
copy sys:system/format rad:system
copy sys:system/diskcopy rad:system
copy sys:system/cli rad:system
copy >nil: sys:shel#? rad:
skip exit
;-------------------------------------------
lab make3
echo "*ec"
sys:system/diskcopy <nil: from df0: to rad: name "RAD"
skip exit
;-------------------------------------------
lab make4
copy >nil: sys:c rad:C
copy >nil: sys:L rad:L
copy >nil: sys:Libs rad:LIBS
copy >nil: sys:System rad:SYSTEM
copy >nil: sys:S rad:S
copy >nil: sys:DEVS rad:DEVS
copy >nil: sys:devs/printers rad:Devs/Printers
copy >nil: sys:devs/keymaps rad:devs/Keymaps
copy >nil: sys:utilities rad:utilities
copy >nil: sys:prefs rad:prefs
copy sys:system.info rad:
copy sys:prefs.info rad:
copy sys:utilities.info rad:
copy sys:disk.info rad:
copy >nil: sys:sh#? rad:
;-------------------------------------------
```

ghCyl, si modifica la dimensione che assumerà la Rad.

Il nostro batch file, però, si propone di operare in completo automatismo, evitando all'utente di dover apportare il minimo intervento. A tal fine, viene in aiuto una opportunità offerta dal comando Mount. Se, infatti, si adopera una forma sintattica...

`Mount Rad: FROM nomefile`

...il sistema cercherà le caratteristiche da assegnare alla Rad non più nel file Mountlist, ma in quello specificato dal parametro "nomefile".

Ecco allora che il nostro batch crea in ram disk un file di nome `Mountrad`, che verrà poi utilizzato nella sintassi di *Mount* con l'opzione *From*. Per creare il file, "gioca" con una serie di redirezioni del comando **Echo**, le quali creano in pratica un file per ogni riga, copiando quanto contenuto nella Mountlist a proposito della entry "Rad:". Tutti questi file (che assumono come nome un numero: 1, 2, 3, eccetera), vengono poi fusi assieme tramite l'uso di **Join** (si consulti anche *Amigafacile* per maggiori ragguagli), creando così un unico file contenente la sola entry Rad. Unico elemento variabile, sarà come ovvio la specifica **`HighCyl`**, che verrà impostata al valore proposto nel menu tramite una serie di condizioni **If... Endif** che testano il contenuto della variabile immessa come input. Il tutto può sembrare un po' complicato... ed in effetti lo è.

Si segua attentamente lo sviluppo iniziale del listato, e prima o poi(!) se ne verrà a capo, aiutandosi con il manuale del Dos e magari con a portata di mano i precedenti numeri di CCC.

Dopo aver "mountato" la Rad, non resta che il lavoro di routine, a seconda della scelta fatta da menu. Per le opzioni **A** e **B**, il file viene subito concluso se si opta per una Rad vuota, altrimenti viene semplicemente copiata la directory C del disco di sistema in Rad, e un opportuno **Assign** farà cercare i comandi eventualmente digitati da Shell in questa nuova allocazione (sezione **Lab make1** del listato).

Leggermente più complessa, invece, la procedura per abilitare una **Rad bootabile di 330 Kb** per configurazioni hardware con 1 Mb di Ram (sezione **Lab make2** del listato). In questo spazio, infatti, andrà inserito quanto necessario al sistema per operare autonomamente,

```
lab exit
echo "*e[33m "
info
echo "*N RAD PRONTA! TOGLI IL FLOPPY"
echo "DA DF0: E RESETTA IL COMPUTER *N*e[0m"
quit
```

senza cioè far ricorso ai dischetti. Il che significa creare le opportune directory in Rad, ed "imbottirle" nei limiti del possibile. Prima di ogni altra cosa, però, è necessario fornire una nuova **startup-sequence**, che tenga conto di quanto potrà trovare nel nuovo sistema interamente in Rad. Anche in questo caso, si adopera la tecnica di redirigere l'output del comando Echo, fondendo poi con Join le varie righe del file Startup-sequence, che verrà ovviamente memorizzato nella directory **S** creata in Rad. Si noti come, sfruttando prima **Ask** e poi ancora l'uso delle variabili di sistema, viene mantenuta la possibilità di optare per una **tastiera** italiana oppure statunitense. Tutte le altre operazioni, altro non sono che una banale serie di **Copy** per trasferire quanto più possibile dal disco Workbench in Rad (naturalmente nelle corrette directory), mantenendo anche l'uso di una eventuale **stampante**. Dopo ogni reset, il sistema rientrerà in ambiente Workbench, mentre dall'icona Rad sarà possibile accedere alla Shell. Chi lo volesse, può modificare la sezione del batch che si occupa di creare la startup-sequence in modo da lasciare attiva la Shell piuttosto che il Workbench: basterà modificare il contenuto dei parametri che seguono Echo esattamente come se si volesse cambiare la startup-sequence del Workbench. Si tenga presente, inoltre, che per l'opzione **C** del batch file Multirad è stata impostata una dimensione di 330 Kb per non creare problemi a chi dispone del Megabyte di memoria diviso in 512K di **Chip** e 512K di **Fast Ram**. Aumentando il valore di HighCyl si potrebbe infatti ottenere una Rad più capiente, ma, oltre all'inevitabile decurtamento della memoria totale, il sistema "divorerebbe" soprattutto la chip ram, creando molti problemi anche nella semplice gestione delle finestre di Intuition. Chi disponesse di 1 Megabyte di Chip Ram, può incrementare con maggiore libertà il valore **29** assegnato ad HighCyl.

Le opzioni per chi dispone di abbondante memoria non necessitano di molte delucidazioni. La più semplice, ed anche la più veloce in fase di creazione, è la quarta (**D**). Il Mount, in questo caso, provvede infatti ad installare una Rad delle stesse dimensioni di un floppy (**80 cilindri**), per cui è possibile adoperare il comando **Diskcopy**, proprio come se si trattasse di due dischetti (sezione **Lab make3** del listato). Nonostante la comparsa (momentanea) di alcuni requester durante le operazioni, la scelta di quest'opzione non richiede comunque altri input da tastiera: basterà il solito reset finale (senza dischi in df0:) per disporre di un sistema completo, ma in Rad.

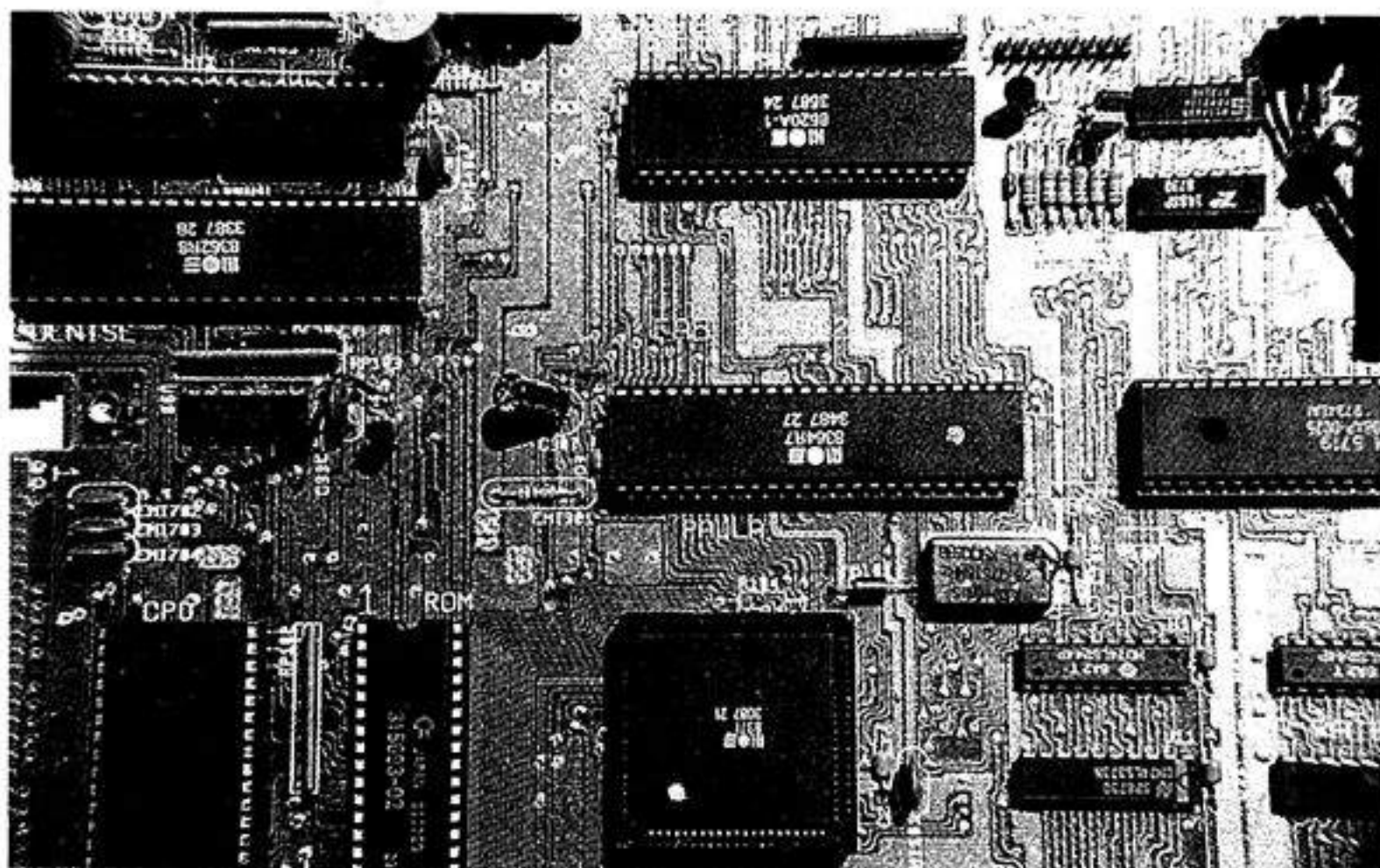
Si tenga presente che, volendo, si può apportare qualsiasi modifica al normale disco Workbench (o meglio: ad una sua **copia**!) e testarla sul floppy. Se funziona, andrà comunque bene anche per la Rad, a meno che non si eliminino file vitali per il sistema (cosa della quale ci si accorgerebbe in ogni caso). Sarà una copia dell'intero disco a costituire il nuovo ambiente.

Dulcis in fundo, l'opzione per chi "sguazza" in **più di due megabyte** di ram. L'uso della Rad, in questo caso, servirà più che altro a supplire l'eventuale **mancanza di hard disk**, per velocizzare le operazioni di reboot e disporre di abbondante spazio per memorizzare senza troppi patemi quei programmi o files che si adoperano con maggiore frequenza. La Rad installata da questa opzione corrisponde alla stessa capienza di 2 floppy, il che però impedisce l'uso di Diskcopy per organizzarne il contenuto.

Si renderà dunque necessario (sezione **Lab make4** del listato) effettuare una serie di Copy tra il disco Workbench ed il nuovo device, che si concluderà (dopo un'attesa più lunga che nei precedenti casi) con il solito invito a resettare il computer dopo l'estrazione del floppy da **df0:**.

In tutte le opzioni, ad installazione avvenuta, viene attivato anche il comando **Info**, che confermerà (se tutto è andato per il meglio) la presenza della Rad, mostrandone anche le sue dimensioni.

Per concludere, non resta da ricordare che la Rad resiste sempre al reset, può essere bootabile solo in presenza di sistema operativo versione 1.3, e che, in caso di lavori importanti, è opportuno salvare una copia di quanto si sta facendo anche su floppy, non solo su Rad. Alcuni Guru particolarmente cruenti riescono infatti (molto di rado, per fortuna) a devastare l'intero sistema, Rad compresa.

Sì alla Recoverable Ram, dunque, ma... la prudenza non è mai troppa.

di Carlo d'Ippolito

Un'affettuosa amicizia tra Amiga e Ms - Dos

Diverse lettere giunte in Redazione dimostrano che non tutti sanno come scambiare dati tra il "mondo" Amiga e quello Ms - Dos; ed altri mondi alieni...

Numerosi utenti di Amiga spesso hanno a che fare con altri utenti che, però, utilizzano sistemi Ms - Dos.

Lo scambio di dati, tra i due mondi, è inoltre definito, sbrigativamente, *impossibile* oppure *totale*.

Si nota tuttavia, soprattutto fra i principianti, un marasma spesso completo, favorito da una scarsa divulgazione di concetti decisamente elementari che, però, tali non sono per chi bazzica poco (o da poco) nel campo del personal computing.

In queste pagine vedremo di chiarire alcuni concetti base, prendendo spunto da un programma, purtroppo non molto noto (o utilizzato) proprio da chi, invece, troverebbe ampi vantaggi nell'usarlo.

Compatibilità vo cercando

Negli ultimi tempi le software house, forti del successo di vendite di Amiga (oltre 2 milioni di esemplari) ed Ms - Dos (molti, molti di più...) hanno deciso di commercializzare gli stessi programmi in entrambe le versioni.

Anche nei giochi più famosi, come ad esempio i simulatori di volo, non si avverte quasi alcuna differenza tra le due versioni; tanto che, se si esamina solo lo schermo durante il loro funzionamento, non si è in grado di stabilire il tipo di computer in quel momento attivato.

La notevole somiglianza esistente tra le due versioni disponibili per lo stesso programma (ciè Amiga ed Ms - Dos), spesso fa ritenere che il supporto magnetico, sul quale è registrato, possa essere "letto" indifferentemente da uno qualsiasi dei due computer.

E' bene quindi chiarire che, sebbene gli *"effetti"* di un programma siano gli stessi, il suo codice, registrato su supporto magnetico, è straordinariamente diverso e non è assolutamente possibile esaminare, con un computer, un dischetto idoneo per funzionare sull'altro.

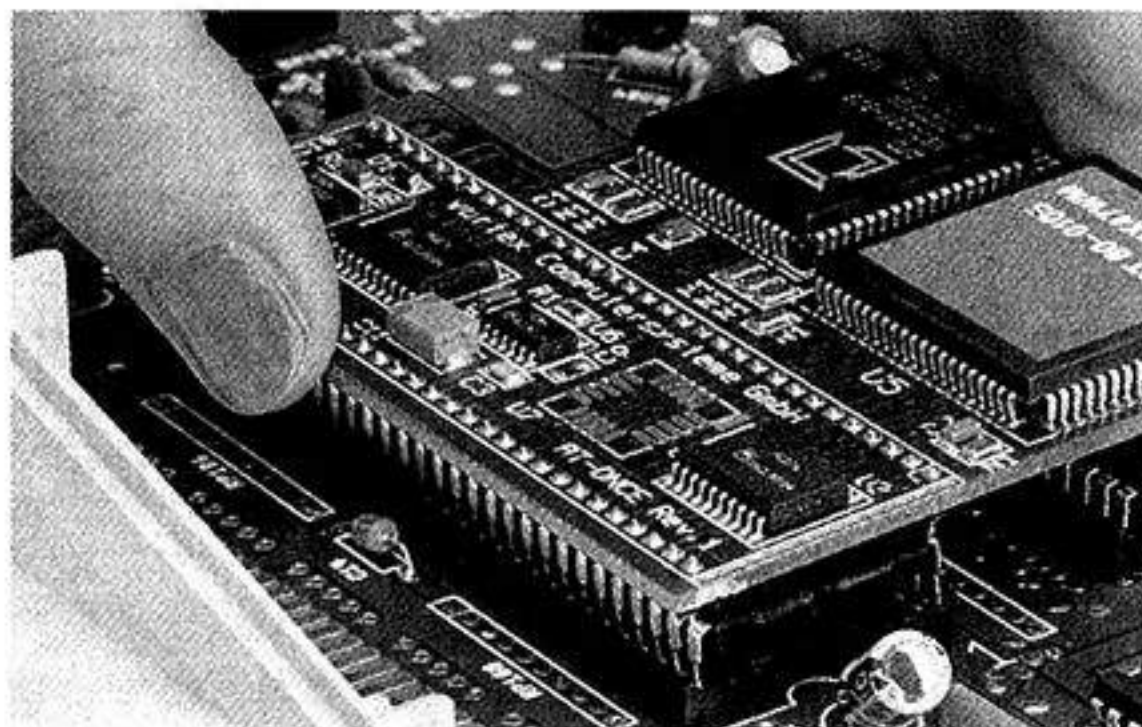
E' come se si pretendesse di utilizzare il gasolio per far funzionare auto a benzina (e viceversa): entrambi i carburanti, è vero, consentono il movimento di una vettura, ma questa deve avere un motore idoneo al tipo di carburante inserito nel serbatoio.

Due computer, quindi, possono offrire lo stesso risultato (= due auto avanzano lungo una stessa strada e, magari, alla stessa velocità) pur se il microprocessore (= il motore) ed il software (= il carburante) sono rigorosamente diversi e non hanno nulla in comune tra loro.

Eccezioni

Qualcuno potrebbe obiettare che è possibile visualizzare, sullo schermo di un Ms - Dos compatibile, una schermata grafica (badate bene: la **stessa** schermata grafica) estratta, non si sa come, dal dischetto di un Amiga. Sono molti che possono giurare di aver visto, magari durante una manifestazione fieristica, esperimenti del genere.

Bisogna quindi chiarire che una cosa è la compatibilità di un **programma**, altra cosa è la compatibilità dei **dischetti**; altra ancora è la compatibilità tra **files** generati da un programma.

E' venuto il momento di spiegarci meglio, partendo proprio da questi ultimi.

Files

Se, operando ad esempio in **Basic**, digitiamo...

```
Print chr$ (65)
```

...vedremo apparire, sullo schermo del computer usato (qualunque esso sia) il carattere "**A**". Al valore **65**, infatti, corrisponde, secondo il codice internazionale **ASCII**, il carattere "A".

E' intuitivo che, se registrassimo su dischetto tale carattere, sarà possibile individuarlo (grazie ad uno delle centinaia di programmi di utilità disponibili) proprio per la presenza del valore 65 in qualche parte del supporto magnetico.

Se, ancora, inviamo via telefono (mediante un **modem**) il carattere-codice 65 ad un nostro amico, questi riceverà la nostra "A", qualunque sia il computer usato. Ciò dimostra che il valore - codice 65 rappresenta sempre e comunque una "A", grazie ad una convenzione internazionale (appunto, il codice Ascii) che facilita lo scambio di dati tra utenti di computer diversi.

E' infatti ovvio che, oltre alla "A", vi sono valori - codice per ciascuna lettera dell'alfabeto (maiuscola e minuscola), per i segni di punteggiatura, per i caratteri numerici (da 0 a 9) e per altri caratteri speciali, sui quali non ci intratteniamo.

Dal momento che una lettera, o un documento in genere, è formato da caratteri alfanumerici, ecco che l'uso dei valori - codice standardizzati Ascii consente lo scambio di documenti (volgarmente detti **files Ascii**) anche tra utenti che utilizzano computer strepitosamente diversi tra loro, purchè, ovvio, rispettino tutti lo standard Ascii.

Riepiloghiamo: se scrivo una lettera con un computer Apple **Macintosh** (che rappresenta uno standard a parte), posso inviarla, via telefono, ad un utente dell'obsoleto sistema **MSX** (addirittura!).

Se tanto mi dà tanto, potrebbe sembrare normale, non disponendo di un modem, inserire in un computer MSX (o Commodore, o di qualsiasi altra marca) il dischetto, estratto da un Macintosh, che contiene la lettera digitata, e memorizzata, servendosi di quest'ultimo computer.

Se operiamo in questo modo, invece, il computer segnala un **errore** di lettura. Come mai?

I dischetti

Una cosa, infatti, è il codice Ascii; altra cosa è, al contrario, il sistema di memorizzazione dei dati.

Ogni fabbricante, per motivi di "prestigio" (o per motivi di copyright) può inventare uno standard di memorizzazione su disco, vale a dire un metodo di archiviazione di dati su supporto magnetico.

Spieghiamoci meglio prendendo in considerazione varie agende. Ve ne sono di tanti tipi: giornaliere, settimanali, tascabili, da tavolo, specifiche per professionisti (sono riportate le date di scadenze particolari), universali (dovete voi scrivere il giorno della settimana), scolastiche (partono da settembre per finire a giugno), mezzo formato, colorate, in carta riciclata, eccetera.

Nonostante il loro scopo sia *identico* per tutte (cioè annotare, con un certo ordine, le cose da fare), la pagina 43 (presa a casaccio, ad esempio) non sarà eguale per tutte le agende.

Supponiamo che sulla nostra agenda sia indicata una certa scadenza il giorno 15 maggio (giorno che corrisponde alla pagina 43 della **nostra** agenda). Se vogliamo ricordare ad un nostro conoscente quella particolare scadenza, non gli suggeriremo di annotarla sulla **sua** agenda (diversa dalla nostra) a pagina 43: questa, infatti, rappresenta una collocazione ben precisa solo sulla nostra agenda. Ci limiteremo a comunicare, invece, la data del 15 maggio: sarà compito del nostro conoscente individuare, nella sua agenda, la pagina idonea in cui annotare l'informazione.

Dunque: i caratteri alfanumerici hanno sempre lo stesso codice Ascii (= data del calendario); ogni computer, invece, ha un suo sistema di memorizzazione (= pagine in cui è suddivisa l'agenda).

Se inviamo, via modem, un documento (= comunichiamo la data di una scadenza), il computer che lo riceve è in grado di memorizzarlo automaticamente sul dischetto (= lo trascrive nella pagina idonea, nel formato confacente alla sua struttura hardware).

Un tentativo di utilizzare la pagina, invece della data (o viceversa) per individuare un'annotazione, genera un errore: un computer ha infatti una struttura piuttosto "rigida" e non è in grado di adattarsi ad altri sistemi di memorizzazione.

I programmi

L'incompatibilità, sulla quale ci soffermiamo non più di tanto, riguarda i programmi. E' fin troppo noto che un programma di word processor che "gira" su Amiga, ad esempio, **non** può girare su un computer Ms - Dos; è come se un cinese tentasse di comunicare con noi nella sua lingua: emette vocali e consonanti (= caratteri di base del codice Ascii), ma la *successione* di queste non ha alcun significato per noi (= la successione dei codici rappresenta una serie di comandi assolutamente inconcludenti per il microprocessore).

Attenzione, però: una cosa è il programma (incompatibile con computer diversi), altra cosa può essere il file generato dal programma.

Abbiamo già visto, infatti, che un documento scritto da un **text editor** che gira su Macintosh, può essere inviato via modem, e correttamente interpretato, da un altro **text editor** di un computer totalmente diverso. Ciò grazie allo standard internazionale Ascii. E' solo una questione, quindi, di mettersi d'accordo sul significato di una **successione** di dati.

Si può, ad esempio, stabilire che una qualsiasi schermata grafica sia formata da una "intestazione" universalmente interpretabile. Le prime due informazioni (memorizzate, come tutte le altre, sempre in codice Ascii) potrebbero rappresentare, nella nostra ipotesi, la risoluzione orizzontale e verticale dello schermo; i successivi bytes, poi, potrebbero rappresentare i pixel (accesi o spenti) memorizzati per righe (o per colonne) di schermo; altri bytes presenti dopo questi potrebbero, sempre nella nostra ipotesi, rappresentare i colori di ciascun pixel visualizzato; e così via.

Ed è proprio quello che un file IFF rappresenta; non proprio come l'abbiamo sommariamente (ed erroneamente) descritto, ma quasi.

I file **IFF** (**I**nterchange **F**ile **F**ormat), infatti, sono memorizzati secondo particolari formati che li rendono idonei ad essere interpretati da un qualsiasi sistema computerizzato su cui giri un programma in grado, ovviamente, di interpretarli.

Ed è questo il motivo per cui una **stessa** schermata grafica, visualizzabile su Amiga, può essere visualizzata anche su un computer Ms - Dos: è sufficiente un

programma in grado di interpretare correttamente il file ricevuto, magari, via modem.

Ed è anche il motivo per cui particolari schermate grafiche di Amiga **non** possono essere visualizzate su un Ms - Dos a causa della presenza di un "modo" grafico incompatibile con quest'ultimo sistema operativo.

Ci riferiamo, in particolare, alle schermate di Amiga in cui sono presenti **4096 colori**, quantità per ora non gestibile dalle comuni schede grafiche Ms - Dos. Attenzione, però: il file di una schermata con 4096 colori può essere in formato IFF; se un Ms - Dos non è in grado di interpretarla non signignifica che il formato IFF è incompatibile, ma solo che il computer considerato non dispone di hardware adeguato.

Allo stesso modo è possibile, con un Macintosh, inviare via modem un documento lunghissimo, magari di un megabyte (un intero libro, ad esempio). Se il computer che riceve il file non dispone di un dischetto adeguato, non si può parlare di *incompatibilità*, ma solo di *inadeguatezza*.

Strane incompatibilità

Capitano casi in cui ciò che abbiamo affermato fino a questo momento sembra non avere alcun valore.

Ci riferiamo, in particolare, a files generati su un computer che, in seguito, sembrano non essere interpretabili dallo **stesso** computer che li ha generati!

Prendiamo, ad esempio, il caso di un word processor; ed esattamente **Wordstar** per sistemi Ms - Dos.

Se, con questo w/p, scrivete un qualsiasi documento, questo verrà correttamente visualizzato su schermo o stampato su carta. Se, però, tentate di leggere il file memorizzato su disco con il noto comando **TYPE** del Dos, vedrete apparire, **oltre** ai caratteri alfanumerici che rappresentano il documento stesso, **altri** caratteri, apparentemente casuali.

Naturalmente, casuali non sono affatto: sono particolari codici di controllo del documento che **rappresentano** (ma solo per il word processor W/s) utili indicazioni per la corretta visualizzazione (su video o stampante) del documento stesso. Siamo, ancòra una volta, in presenza di un *sistema di codifica*, pur se, stavolta, *non universale* come quello Ascii; è diverso dal codice Ascii, ma sempre di codifica si tratta!

Ecco, dunque, il motivo per cui un documento memorizzato con W/s può essere correttamente interpretato e gestito **solo** da W/s. Il comando Type, invece, considera i codici di controllo W/s come normali codici Ascii e come tali li visualizza.

Allo stesso modo, alcuni programmi sono in grado di gestire **esponenti** e **deponenti** ma, per svolgere tale compito, sono "costretti" ad usare specifici codici-comandi, non individuabili da altri programmi.

E' lo stesso motivo per cui alcuni documenti, in cui compaiono **vocali accentate**, ne sembrano privi (oppure vengono visualizzati, al loro posto, caratteri privi di significato) se "trattati" da *programmi diversi* da quello che li ha generati.

Il problema, ovviamente, è comune non solo ai w/p, ma anche ai **data base** ed ai **fogli elettronici**; ed è un problema che, inoltre, si presenta su *qualsiasi* sistema computerizzato.

Traduzioni

Per evitare confusione, i produttori di computer (e di software) avrebbero potuto accordarsi, una volta per tutte, per usare codici realmente universali.

La tradizionale gelosia imperante tra i costruttori ha però consentito (almeno questo!) di avere in comune **solo** il codice Ascii (e neppure in modo univoco...).

Per evitare inconvenienti agli utenti, tuttavia, alcune software house offrono, nei programmi più recenti, la possibilità sia di leggere files generati da programmi analoghi, ma offerti dalla "concorrenza"; sia, viceversa, di memorizzare un file in formati "diversi".

Così, ad esempio, il popolare foglio elettronico **Quattro**, della **Borland**, è in grado di leggere (e scrivere) files in formato **123** (tradizionalmente concorrenziale nel campo dei fogli elettronici) oltre che, ovviamente, nel formato Borland.

Allo stesso modo, il pacchetto di impaginazione **Ventura** è in grado di leggere files provenienti dai più diffusi w/p (tra cui Ms-Word, Word Star, Writer, Wordperfect, Multimate ed altri) oltre che semplici e "banali" files Ascii.

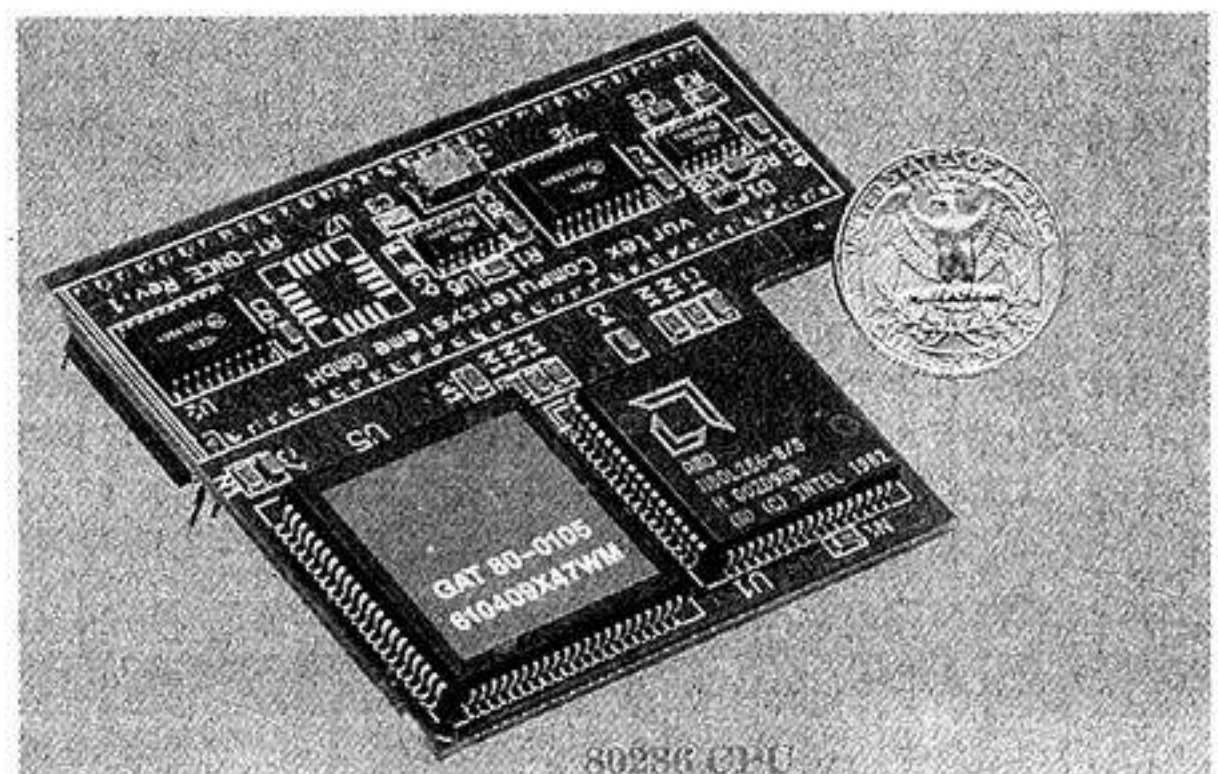
Non tutti gli utenti di W/s, ad esempio, sanno che è possibile, da un file W/s, eliminare tutti i caratteri di controllo ivi contenuti allo scopo di trasformare, appunto, un file W/s in un comune file Ascii interpretabile correttamente perfino dal comando Type.

Il programma (**WsAscii**), di pubblico dominio, è lungo poche migliaia di bytes e funziona nel modo più banale che sia possibile immaginare: richiede il nome del file W/s da "convertire" ed il nome da assegnare al file convertito.

Alla fine delle operazioni saranno presenti **due** files: quello originale (il primo) e quello "depurato" dai codici di controllo tipici di W/s (il secondo).

Da un disco all'altro

Sembrerebbe, quindi, che lo scambio di files di tipo "universale" (files Ascii, IFF ed altri che seguono standard noti) sia possibile solo via modem dal momento che (l'abbiamo affermato in un precedente paragrafo) non è possibile, con un determinato elaboratore, gestire dischetti provenienti da computer diversi.

In *teoria* dovrebbe esser così, ma alcune circostanze fortunose hanno consentito lo sviluppo di particolari procedure che aggirano l'ostacolo.

In che cosa può differire un sistema di memorizzazione da un altro?

Si tratta, in fin dei conti, di un dischetto magnetico che gira, sul quale una testina legge o scrive (a seconda dei casi) dei dati.

Se, quindi, la velocità di rotazione è uguale (ed è proprio il caso di Amiga ed Ms - Dos) dovrebbe esser possibile memorizzare informazioni allo stesso modo su entrambi i computer.

Nel caso di elaboratori Ms - Dos, tuttavia, la gestione della testina di lettura scrittura è affidata ad alcuni componenti **hardware** presenti sulla scheda del drive; ne consegue che l'alterazione della procedura è impossibile.

In Amiga, invece, tale gestione è affidata al **software** della macchina: nei drive Amiga sono infatti presenti solamente (o quasi) le circuiterie necessarie per far girare i dischetti.

Ad alcuni programmatori, quindi, è venuta in mente l'idea di scrivere un programma di gestione del disco che simula in modo completo la manipolazione dei dati (a livello hardware!) sulla superficie del dischetto magnetico.

Uno di questi programmi, il notissimo **Dos 2 Dos**, realizza quindi una completa compatibilità hardware che consente, ad **utenti Amiga**, di leggere (o scrivere) files da (su) dischetti provenienti dall'ambiente Ms - Dos. Il viceversa **non** è possibile, per i motivi prima descritti.

Ribadiamo ancora una volta (ma a questo punto dovrebbe esser superfluo) che la possibilità di leggere, con Amiga, dischetti Ms - Dos **non** consente, ad Amiga, di far **girare** programmi Ms - Dos. E' "solo" possibile lo scambio di files "universali" (Ascii, IFF, e così via); e non diteci che è poco...

Dos 2 Dos

Il programma, lungo circa 30 K bytes, è noto agli appassionati in varie "vesti": dotato (o privo) di icona, dotato (o privo) di requester, variamente colorato e così via.

Il programma ridotto all'osso, però, una volta lanciato (semplicemente digitandone il nome da Shell) chiede di indicare il drive che deve "ospitare" il dischetto in formato Ms - Dos. Chi dispone di un solo drive, ovviamente, digiterà **DF0:** e premerà return; si può rispondere, comunque, indicando un qualsiasi drive (da DF0: a DF3:).

L'ideale, ovviamente, è quello di disporre di **due** drives: nel primo si inserirà il dischetto in formato Amiga, nel secondo quello Ms - Dos. Non dimentichiamo, infatti, che scopo principale del programma Dos 2 Dos è quello di **trasferire** files da un ambiente all'altro allo scopo di trattarli in seguito, con comodo, dopo aver effettuato il trasferimento.

Disponendo di un solo drive le operazioni da compiere si allungano un po'. Nel caso di desideri riversare files Amiga in un dischetto Ms - Dos è infatti necessario, **prima** di lanciare Dos 2 Dos, caricare in **RAM:** tutti i files da trasferire e, in seguito, trasferirli. Nel caso ci si accorga di aver dimenticato qualche file è indispensabile "uscire" dal programma (il drive, infatti, è ora abilitato a leggere dischetti in formato Ms - Dos e non più Amiga) e ripetere la procedura.

Chi dispone di due drives, invece, in qualsiasi momento potrà disporre di entrambi i formati.

Nel caso contrario (trasferimento da Ms - Dos ad Amiga) sarà possibile inserire tutti i dischetti Ms - Dos che si desidera ed effettuare trasferimenti in RAM: fino a riempire l'intera memoria disponibile. Al termine delle operazioni, usciti da Dos 2 Dos, sarà possibile riversare il contenuto della Ram: in dischetti Amiga.

Opzioni

Le opzioni disponibili da Dos 2 Dos sono numerose, qui di seguito commentate una per una.

Dir. Consente di esaminare la directory di qualsiasi drive attivo e della Ram: Volendo, si può indicare il path (percorso) che va digitato prestando particolare attenzione all'inclinazione della barra che indica le subdirectory: l'inclinazione che indica il path è infatti diversa tra Amiga (/) ed Ms - Dos (\).

Chdir. Rende attiva una particolare subdirectory; opzione comoda nel caso si debbano trattare files nidificati all'interno di numerose subdirectory.

Type. E' il ben noto comando Dos, utile per stabilire se un certo file è in formato Ascii (e quindi un suo traferimento può essere utile) oppure se è un programma (in questo caso il trasferimento, benchè possibile, sarebbe inutile).

Copy. Inutile commentare questo indispensabile comando, che è il motivo principale dell'esistenza di Dos 2 Dos. Si noti che è possibile usare i caratteri **wildcards**, cioè l'asterisco (*****) per il mondo Ms - Dos e il cancelletto e punto di domanda (**#?**) per quello Amiga. Nel caso in cui il nome di un file Amiga risultasse incompatibile con lo standard Ms - Dos (ad esempio, se fosse di lughezza maggiore di 8 caratteri) l'operazione di copia viene interrotta ed il programma chiede di indicare un nome alternativo. L'operazione si interrompe egualmente (con richiesta di conferma) nel caso in cui il nome del file da copiare sia già presente nel dischetto destinazione (onde evitare indesiderate cancellature).

Delete. Comando ovvio che non merita commento.

Format. Con questo comando è possibile inserire un dischetto nel drive di Amiga e formattarlo nel formato Ms - Dos (720 K byte), pur se con una velocità minore rispetto ad una formattazione realizzabile su un "vero" computer Ms - Dos.

X. Consente il "ritorno" nell'ambiente Shell di Amiga.

Una particolare attenzione merita il comando Copy che offre due forme sintattiche. Con la prima (digitando, in "coda" al comando, i due caratteri **-r**) si evita che il computer chieda conferma nel caso vi siano due files di nome eguale.

La seconda forma sintattica, invece, si realizza digitando **-a** in "coda" al comando stesso. In questo modo, se in un file Amiga vengono individuati caratteri Ascii di codice 10, questi vengono "trasformati" nella **coppia** di caratteri Ascii **13 + 10**.

Non tutti sanno, infatti, che moltissimi programmi Amiga generano files che codificano il carattere **Return** con il solo codice Ascii 10. Nel mondo Ms - Dos, invece, il carattere Return viene individuato dalla coppia di caratteri 13 (ritorno carrello) e 10 (avanzamento di una riga). In alcuni casi, tra cui il trasferimento di files di testo, tale opzione sarà particolarmente utile per evitare il ricorso a specifci programmi di adattamento.

GUIDA MERCATO

AMIGA

Amiga 3000 /40 - L. 5.200.000
Microprocessore Motorola MC68030 - Amiga 3000 a 16 Mhz - Floppy Disk Drive 3 1/2" da 880 Kb - Hard Disk da 40 Mb

A3025/40 - L. 6.200.000
Stessa configurazione del A3016/40 ma con clock a 25 Mhz

A3025/100 - L. 7.050.000
Stessa configurazione del A3016/40 ma con clock a 25 Mhz e Hard Disk da 100 Mb

Amiga 2000 - L. 1.484.000
Microprocessore Motorola MC68000 - Clock 7.16 Mhz - 1 Mbyte RAM - 1 Floppy Disk Drive da 3 1/2", 880 Kbyte

Amiga 500 - L. 786.000
Microprocessore Motorola MC68000 - Clock 7.16 Mhz - 512 Kbyte RAM - 1 Floppy Disk Driver da 3 1/2", 880 Kbyte

A500 FUNLAB - L. 1.260.000
Kit Amiga 500 + tastiera KAWAI + software STEINBERG

C 64

CPU C64C - L. 256.000
Nuovo Personal Computer C64 - 64 Kb RAM - Vastissima biblioteca software disponibile - Chip custom specializzatPio per audio e video - rta seriale Commodore - Porta registratore per cassette - Porta parallela programmabile

C64 GS - L. 180.000
ConfigurazNiuoone C64 Game System

KIT SCUOLA - L. 334.000
Nuovo C64C Scuola

SKATE - L. 347.000
Kit Skateboard

Drive 1541 Kit - L. 280.000
Floppy disk drive 5 1/4" singola faccia - Capacita' 170 Kb Formattati - Compatibile con C64, C128, C128D

1530 - L. 44.000
Registratore a cassette per C64, C128, C128D - Utilizza normali cassette audio per la memorizzazione di programmi e dati

128M - L. 39.000
Mouse 1350 per C64, C128, C128D - Adatto per applicazioni grafiche e per l'ambiente GEOS

STAMPANTI

MPS 1230 - L. 291.000
Stampante a matrice di punti - 9 aghi - 80 colonne - Compatibile con tutti i prodotti Commodore

MPS 1270 - L. 444.000
Stampante a getto d'inchiostro - 80 colonne - Interfaccia Parallela Centronics

MPS 1550C - L. 363.000
Stampante a colori - 9 aghi - 80 colonne di stampa - Compatibile con tutti i prodotti Commodore

MPS 1224C - L. 950.000
Stampante a matrice di punti ad alta qualita' - 24 aghi - 132 colonne - Interfaccia parallela e seriale

MS-DOS

PC 10 10 - L. 709.000
Microprocessore Intel 8088 - Clock 4.77/9.54 Mhz - 640 Kb RAM - 1 Drive 5 1/4" da 360 Kb

PC 10 3 - L. 848.000
Stessa configurazione del PC 10-10 ma con due drive 5 1/4" da 360 Kb

PC 10 FD1 - L. 709.000
Stessa configurazione del PC 10-10 ma con un drive 3 1/2" da 720 Kb

PC 10 FD2 - L. 848.000
Stessa configurazione del PC 10-10 ma con due drive 3 1/2" da 720 Kb

PC 20 3 - L. 1.195.000
Stessa configurazione del PC 10-10 ma con Hard Disk da 20 Mb

PC 20 3 3-2 - L. 1.195.000
Stessa configurazione del PC 10-10 ma con Hard disk da 20 Mb e drive 3 1/2" 720 Kb

PC 30 3 - L. 1.655.000
Microprocessore Intel 80286 - Clock 6/10 Mhz - 1 drive 3 1/2" da 1.44 Mb - Hard Disk 20 Mb - Case Baby

PC 30V 3-2 - L. 1.795.000
Stessa configurazione del PC 30-3 con, in aggiunta, la Super VGA

PC 40 3 1-0 - L. 2.350.000
Microprocessore Intel 80286 - Clock 6/10 Mhz - 1 drive 3 1/2" da 1.44 Mb - Hard Disk 20 Mb

PC 40/40 3 - L. 2.750.000
Stessa configurazione del PC 40-3 1-0 ma con Hard Disk da 40 Mb

PC 50-2 3-0 - L. 3.650.000
Microprocessore a 32 bit Intel 80386sx - Clock a 16 Mhz - 1 Drive 3 1/2" da 1.44 Mb

PC 50-2 3-4 - L. 3.984.000
Come il PC 50-2 3-0 ma con HD 40 Mb

PC 50-2 3-10 - L. 4.750.000
Stessa configurazione del PC 50-2 3-0 ma con Hard Disk da 100 Mb

PC 60-3 5-0 - L. 7.500.000
Microprocessore a 32 bit Intel 80386DX Configurazione base comprendente un Drive 5 1/4" da 1.2 Mb e uno 3 1/2" da 1.44 Mb

PC 60-3 5-8 - L. 7.950.000
Stessa configurazione del PC 60-3 5-0 ma con Hard Disk da 80 Mb

PC 60-3 5-20 - L. 10.200.000
Stessa configurazione del PC 60-3 5-0 ma con Hard Disk da 200 Mb

C286-LT - L. 3.950.000
Notebook con processore Intel 80C286 Configurazione con Drive 3 1/2" da 1.44 Mb
Dotato di Hard Disk da 20 Mb

ACCESSORI

Accessori per MS-DOS compatibili

1352 - L. 75.000
Mouse Microsoft Bus Mouse compatibile - Collegabile direttamente alla serie PC

PC910 - L. 167.000
Floppy Disk Drive interno aggiuntivo da 3 1/2" per la serie PC - Capacita' 720 Kb

PC915 - L. 193.000
Come PC910 ma con capacita' da 1.44 Mb

PC920 - L. 167.000
Floppy Disk Drive interno aggiuntivo da 5 1/4" per la serie PC - Capacita' 360 Kb

PC925 - L. 193.000
Come PC920 ma con capacita' da 1.2 Mb

980 - L. 1.500.000
Hard Disk 80 Mb

9200 - L. 3.500.000
Hard Disk 200 Mb

PC970 - L. 625.000
Streaming Tape 40.6 Mb

PC971 - L. 1.200.000
Streaming Tape 120 Mb

MEM 1/40 - L. 750.000
Espansione 1 Mb per PC 40-3

MEM 2/50 - L. 850.000
Espansione 2 Mb per PC 50-2

PC968-4 - L. 1.850.000
Espansione di memroia da 4 Megabyte compatibile con PC 60-3

PC968-8 - L. 3.100.000
Espansione di memoria da 8 Megabyte compatibile con PC 60-3

Monitor 1402 - L. 125.000
Monitor a fosfori "bianco-carta" - Tubo 12" antiriflesso - Ingresso TTL - Compatibile con tutta la gamma PC

Monitor 1403 - L. 260.000
Monitor monocromatico in standard VGA

Monitor 1404 - L. 168.000
Monitor 14" a fosfori ambra - Compatibile con tutta la gamma PC

Monitor a colori 1930 - L. 880.000
Monitor ad alta risoluzione in standard VGA

Monitor a colori 1950 - L. 1.000.000
Monitor 14" BI-SYNC alta risoluzione - Compatibile con MDA, Hercules, CGA, EGA e VGA

Monitor a colori 1951 - L. 970.000
Monitor a colori VGA (Hitachi)

Accessori per Amiga

A2000TAST - L. 190.000
Tastiera per Amiga 2000

A2000DTV - L. 2.988.000
Desk Top Video per Amiga 2000

A2000DTM - L. 2.792.000
Desk Top Music per Amiga 2000

Drive A1011 - L. 171.000
Drive esterno da 3 1/2" - Capacità 880 Kbyte

Driver A2010 - L. 171.000
Drive interno aggiuntivo da 3 1/2" - Capacità 880 Kbyte - Collegabile ad Amiga 2000

Hard Disk A590 - L. 760.000
Scheda controller - Hard Disk da 3 1/2" 20 Mb - 2 Mb "fast" RAM - Collegabile all'Amiga 500

Scheda Janus A2088 - L. 674.000
Scheda Janus XT - Scheda Bridgeboard per compatibilità MS-Dos (AT) in Amiga 2000 - Clock 8 Mhz - 1 Mb RAM - Floppy Disk Controller - Floppy Disk Drive disegnato per l'installazione all'interno dell'Amiga 2000

Scheda Janus A2286 - L. 1.378.000
Stessa configurazione della scheda A2088 ma con microprocessore Intel 80286

A2630 - L. 3.020.000
Scheda Processore alternativo per 68030 e Unix - Microprocessore Motorola MC68030 - 2 Mb RAM aggiuntivi

A2630/4 - L. 4.074.000
Stessa configurazione della scheda A2630 ma con 4 Mb di RAM aggiuntivi

A2032 - L. 127.000
Modulatore PAL per Amiga

A2320 - L. 460.000
De-Interlacer - Flicker Fixer per Amiga

A2091 - L. 288.000
Controller A2091 per Amiga 2000

A2091/40 - L. 1.194.000
Controller A2091 + Hard Disk da 40 Mb per Amiga 2000

A2058 - L. 798.000
Scheda di espansione per Amiga 2000 fornita con 2 Mb "fast" RAM, espandibile a 4 o 8 Mb

A2300 - L. 298.000
Scheda Genlock semiprofessionale per Amiga 2000

A501 - L. 171.000
Cartuccia di espansione di memoria da 512 Kb per Amiga 500

A520 - L. 42.000
Modulatore RF esterno per Amiga 500 - Permette di connettere qualsiasi televisore all'Amiga 500

Monitor a colori 1084 - L. 472.000
Monitor a colori ad alta risoluzione - Tubo 14" antiriflesso - Compatibile con Amiga 500/2000, PC (tutta la gamma), C64 e C128

Monitor A2024 - L. 1.055.000
Monitor Monocromatico a fosfori "bianco-carta" - Tubo 14" antiriflesso

A2060 - L. 450.000
Scheda ARCNET per Amiga

A2065 - L. 590.000
Scheda ETHERNET per Amiga

Altoparlanti A10 - L. 60.000
Set di amplificazione per l'uscita audio di Amiga

Studio Bitplane - Software per corrispondenza

Il valore dell'utility al costo del videogame

Richiedi il catalogo gratuito!

Il C64 supera se stesso e vola anni luce avanti!

Quando si possiede un ottimo computer, quello che serve è dell'ottimo software, utile e stimolante per chi non vuole solo giocare. Per gli appassionati dell'immortale Commodore 64 offriamo software ai vertici della programmazione e dell'utilità: package per la compilazione, archiviazione e stampa di fatture, gestione magazzino (entrate/uscite, prezzi, variazioni, saldi), creazione e stampa personalizzata di grafici (a barre, a torta, a linee, multipli, 2D e 3D), desktop video (titolazione di videocassette, presentazione di programmi, effetti audio/video), archiviazione dati, programmazione, grafica, musica, file, ecc. Tutto il software, che sarebbe ben poca cosa senza chiare e semplici istruzioni per l'uso, include le istruzioni in italiano e costa meno di un videogame!

Amiga scopre il free software in italiano

Per gli utenti Amiga raccogliamo e selezioniamo free software da tutto il mondo (migliaia di programmi di ogni tipo, dal potente spreadsheet alla micro-routine per gestire il joystick in assembler!), integrandolo con istruzioni in italiano!!

Per ricevere i cataloghi gratuiti inviate il vostro indirizzo a:

Studio Bitplane
casella postale 10942
20124 Milano

Studio Bitplane - cas. post. 10942 - 20124 Milano - tel. 02/39320732 - vendita per corrispondenza, spedizioni in tutta Italia

I COMMODORE POINT

LOMBARDIA

Milano
• AL RISPARMIO - V.LE MONZA 204
• BCS - VIA MONTEGANI 11
• BRAHA A. - VIA PIER CAPPONI 5
• E.D.S. - C.SO PORTA TICINESE 4
• FAREF - VIA A. VOLTA 21
• FLOPPERIA - V.LE MONTENERO 31
• GBC - VIA CANTONI 7 - VIA PETRELLA 6
• GIGLIONI - V.LE LUIGI STURZO 45
• L'UFFICIO 2000 - VIA RIPAMONTI 213
• LOGITEK - VIA GOLGI 60
• LU - MEN - VIA SANTA MONICA 3
• MARCUCCI - VIA F.LLI BRONZETTI 37
• MELCHIONI - VIA P. COLLETTA 37
• MESSAGGERIE MUSICALI - GALLERIA DEL CORSO 2
• NEWEL - VIA MAC MAHON 75
• PANCOMMERZ ITALIA - VIA PADOVA 1
• SUPERGAMES - VIA VITRUVIO 38
• 68000 E DINTORNI - VIA WASHINGTON 91
Provincia di Milano
• GINO FERRARI CENTRO HI-FI - VIA MADRE CABRINI 44 - S. ANG. LODIGIANO
• F.LLI GALIMBERTI - VIA NAZIONALE DEI GIOVI 28/36 - BARLASSINA
• TECNOLUX - VIA PIETRO NENNI 5 - BERNATE TICINO
• OGGIONI & C. - VIA DANTE CESANA 27 - CARATE BRIANZA
• AL RISPARMIO - VIA U. GIORDANO 57 - CINISELLO BALSAMO
• GBC - V.LE MATTEOTTI 66 - CINISELLO BALSAMO
• CASA DELLA MUSICA - VIA INDIPENDENZA 21 - COLOGNO MONZESE
• PENATI - VIA VERDI 28/30 - CORBETTA
• EPM SYSTEM - V.LE ITALIA 12 - CORSICO
• P.G. OSTELLARI - VIA MILANO 300 - DESIO
• CENTRO COMPUTER PANDOLFI - VIA CORRIDONI 18 - LEGNANO
• COMPUTEAM - VIA VECELLIO 41 - LISSONE
• M.B.M. - C.SO ROMA 112 - LODI
• L'AMICO DEL COMPUTER - VIA CASTELLINI 27 - MELEGNANO
• BIT 84 - VIA ITALIA 4 - MONZA
• IL CURSORE - VIA CAMPO DEI FIORI 35 - NOVATE MIL.
• I.C.O. - VIA DEI TIGLI 14 - OPERA
• R & C ELGRA - VIA SAN MARTINO 13 - PALAZZOLO MIL.
• ESSEGIEMME SISTEMI SAS - VIA DE AMICIS 24 - RHO
• TECNO - CENTRO - VIA BARACCA 2 - SEREGNO
• NIWA HARD&SOFT - VIA B. BUOZZI 94 - SESTO SAN GIOV.
• COMPUTER SHOP - VIA CONFALONIERI 35 - VILLASANTA
• ACTE - VIA B. CREMIGNANI 13 - VIMERCATE
• IL COMPUTER SERVICE SHOP - VIA PADANA SUPERIORE 197 - VIMODRONE
Bergamo
• D.R.B. - VIA BORGO PALAZZO 65
• TINTORI ENRICO &C. - VIA BROSETA 1
• VIDEO IMMAGINE - VIA CARDUCCI c/o CITTA' DI MERCATO
Provincia di Bergamo
BERTULEZZI GIOVANNI - VIA FANTONI 48 - ALZANO LOMBARDO
• COMPUTER SHOP - VIA VITTORIO VENETO 9 - CAPRIATE SAN GERVASIO
• B M R - VIA BUTTARO 4/T - DALMINE
• MEGABYTE 2 - VIA ROMA 61/A - GRUMELLO
• OTTICO OPTOMETRISTA ROVETTA - P.ZZA GARIBALDI 6 - LOVERE
• COMPUTER POINT - VIA LANTIERI 52 - SARNICO
• A.B. INFORMATICA - STRADA STATALE CREMASCA 66 - URGNANO

Brescia
• MASTER INFORMATICA - VIA F.LLI UGONI 10/B
PROVINCIA DI BRESCIA
• MISTER BIT - VIA MAZZINI 70 - BRENO
• CAVALLI PIETRO - VIA 10 GIORNATE 14 BIS - CASTREZZATO
• VIETTI GIUSEPPE - VIA MILANO 1/B - CHIARI
• MEGABYTE - P.ZZA MALUEZZI 14 - DESENZANO DEL GARDA
• BARESI RINO &C. - VIA XX SETTEMBRE 7 - GHEDI
• INFO CAM - VIA PROVINCIALE 3 - GRATACASOLO
• "PAC-LAND" di GARDONI - CENTRO COM.le - LA CASA DI MARGHERITA D'ESTE - VIA GIORGIONI 21
Como
• IL COMPUTER - VIA INDIPENDENZA 90
• 2M ELETTRONICA - VIA SACCO 3
Provincia di Como
• ELTRON - VIA IV NOVEMBRE 1 - BARZANO'
• DATA FOUND - VIA A. VOLTA 4 - ERBA
• CIMA ELETTRONICA - VIA L. DA VINCI 7 - LECCO
• FUMAGALLI - VIA CAIROLI 48 - LECCO
• RIGHI ELETTRONICA - VIA G. LEOPARDI 26 - OLGIATE COMASCO
Cremona
• MONDO COMPUTER - VIA GIUSEPPINA 11/B
• PRISMA - VIA BUOSO DA DOVARA 8
• TELCO - P.ZZA MARCONI 2/A
Provincia di Cremona
• ELCOM - VIA IV NOVEMBRE 56/58 - CREMA
• EUROELETTRONICA - VIA XX SETTEMBRE 92/A - CREMA
Mantova
• COMPUTER CANOSSA - GAL. FERRI 7
• 32 BIT - VIA C. BATTISTI 14
• ELET. di BASSO - V.LE RISORGIMENTO 69
Provincia di Mantova
• CLICK - ON COMPUTER - S.S. GOITESE 168 - GOITO
Pavia
• POLIWARE - C.SO C. ALBERTO 76
• SENNA GIANFRANCO - VIA CALCHI 5
Provincia di Pavia
• A. FERRARI - C.SO CAVOUR 57 - MORTARA
• LOGICA MAINT - V.LE M.TE GRAPPA 32 - VIGEVANO
• M. VISENTIN - C.SO V. EMANUELE 76 - VIGEVANO
Sondrio
• CIPOLLA MAURO - VIA TREMOGGE 25
Provincia di Sondrio
• FOTONOVA - VIA VALERIANA 1 - S.PIETRO DI BERBENNO
Varese
• ELLE - EFFE - VIA GOLDONI 35
• IL C.TRO ELET. - VIA MORAZZONE 2
• SUPERGAMES - VIA CARROBBIO 13
Provincia di Varese
• BUSTO BIT - VIA GAVINANA 17-BUSTO A.
• MASTER PIX - VIA S.MICHELE 3 - BUSTO A.
• PUNTO UFFICIO - VIA R.SANZIO 8 - GALLARATE
• GRANDI MAGAZZINI BOSSI - VIA CLERICI 196 - GERENZANO
• J.A.C. - C.so MATTEOTTI 38 - SESTO C.

PIEMONTE

Alessandria
• BIT MICRO - VIA MAZZINI 102
• SERV. INFOR. - VIA ALESSANDRO III 47
Provincia di Alessandria
• SONY ITALIANA - VIA G. MANARA 7 - CASALE MONFERRATO
• SGE ELETTRONICA - VIA BANDELLO 19 - TORTONA
• COMPUTER TEMPLE - VIA F. CAVALLOTTI 13 - VALENZA
Asti
• ASTI GAMES - C.SO ALFIERI 26
• RECORD - C.SO ALFIERI 166/3 (Galleria Argenta)
Cuneo
• ROSSI COMPUTERS - C.SO NIZZA 42
Provincia di Cuneo
• PUNTO BIT - C.SO LANGHE 26/C - ALBA
• BOSETTI - VIA ROMA 149 - FOSSANO
• COMPUTERLAND - VIA MAZZINI 30/32 - SALUZZO
Novara
• PROGRAMMA 3 - V.LE BUONARROTI 8
• PUNTO VIDEO - C.so RISORGIMENTO 39/B
Provincia di Novara
• COMPUTER - VIA MONTE ZEDA 4 - ARONA
• ALL COMPUTER - C.SO GARIBALDI 106 - BORGOMANERO
• S.P.A. - C.SO DISSEGNA 21/BIS - DOMODOSSOLA
• ELLIOTT COMPUTER SHOP - VIA DON MINZONI 32 - INTRA
• TRISCONI VALERIA - VIA MAZZINI 90 - OMEGNA
Torino
• ABA ELETTRONICA - VIA C. FOSSATI 5/P
• ALEX COMPUTER E GIOCHI - C.SO FRANCIA 333/4
• COMPUTER HOME - VIA SAN DONATO 46/D
• COMPUTING NEW - VIA M. POLO 40/E
• C.D.M. ELETTR. - VIA MAROCHETTI 17
• DE BUG - C.SO V. EMANUELE II 22
• DESME UNIVERSAL - VIA S.SECONDO 95
• FDS ALTERIO - VIA BORGARO 86/D
• IL COMPUTER - VIA N. FABRIZI 126
• MICRONTEL - C.SO D. degli ABRUZZI 28
• PLAY GAMES SHOP - VIA C. ALBERTO 39/E
• RADIO TV MIRAFIORI - C.SO UNIONE SOVIETICA 381
• SMT ELETTRONICA - VIA BIBIANA 83/bis
Provincia di Torino
• PAUL E CHICO VIDEOSOUND - VIA V.EMANUELE 52 - CHIERI
• BIT INFORMATICA - VIA V. EMANUELE 154 - CIRIE'
• HI - FI CLUB - C.SO FRANCIA 92C - COLLEGNO
• MISTER PERSONAL - VIA CATTANEO 52 - FAVRIA
• I.C.S. - VIA TORINO 73 - IVREA
• DAG - VIA I MAGGIO 40 - LUSERNA S. GIOVANNI
• EUREX - C.SO INDIPENDENZA 5 - RIVAROLO CANAVESE
• DIAM INFORMATICA - C.SO FRANCIA 146 bis - RIVOLI
• FULLINFORMATICA - VIA V.VENETO 25 - RIVOLI
• GAMMA COMPUTER - VIA CAVOUR 3A-3B - SET.TORINESE
Vercelli
• ELETTROGAMMA - C.SO BORMIDA 27 ang. V.Montanara
• ELETTRONICA - STRADA TORINO 15
Provincia di Vercelli
• C.S.I. TEOREMA - VIA LOSANA 9 - BIELLA
• SIGEST - VIA BERTODANO 8 - BIELLA
• REMONDINO FRANCO - VIA ROMA 5 - BORGOSESIA
• FOTOSTUDIO TREVISAN - VIA XXV APRILE 24/B - COSSATO
• STUDIO FOTOGRAFICO IMARISIO - P.ZZA M. LIBERTA' 7 - TRINO

VENETO

Belluno
• UP TO DATE - VIA V. VENETO 43
Provincia di Belluno
• GUERRA COMPUTERS - V.LE MAZZINI 10/A - FELTRE
Padova
• BIT SHOP - VIA CAIROLI 11
• COMPUMANIA - VIA T. CAMPOSANPIERO 37
• D.P.R. DE PRATO R. - V.LO LOMBARDO 4
• G.F. MARCATO - VIA MADONNA DELLA SALUTE 51/53
• SARTO COMPUTER - VIA ARMISTIZIO 79
Provincia di Padova
• COMPUTER SERVICE - BORGO TREVISO 150 - CITTADELLA
Treviso
• BIT 2000 - VIA BRANDOLINI D'ADDA 14
• GUERRA EGIDIO & C. - V.LE CAIROLI 95
Provincia di Treviso
• DE MARIN COMPUTERS - VIA MATTEOTTI 142 - CONEGLIANO
• SIDESTREET - VIA SALVO D'ACQUISTO 8 - MONTEBELLUNA
• FALCON ELETTROAUDIOVIDEO - VIA TERRAGGIO 116 - PREGANZIOL
Venezia
• GUERRA EGIDIO & C. - VIA BISSUOLA 20/A - MESTRE
• TELERADIO FUGA - SAN MARCO 3457
Provincia di Venezia
• GUERRA EGIDIO & C. - VIA VIZZOTTO 29 - SAN DONA' DI PIAVE
• REBEL - VIA F. CRISPI 10 - SAN DONA' DI PIAVE
Verona
• CASA DELLA RADIO - VIA CAIROLI 10
• TELESAT - VIA VASCO DE GAMA 8
• **Provincia di Verona**
• UBER - CP 0363(RAG.SOC. DERTA) - VIA MASCAGNI 31 - CASTEL D'AZZANO
• FERRARIN - VIA DEI MASSARI 10 - LEGNAGO
• COMPUTERS CENTER - VIA CANTORE 26 - VILLAFRANCA
Vicenza
• ELET. BISELLO - V.LE TRIESTE 427/429
• SCALCHI MARKET - VIA CA' BALBI 139
Provincia di Vicenza
• SCHIAVOTTO - VIA ZANELLA 21 - CAVAZZALE
• GUERRA E. & C. - V.LE DELLE INDUSTRIE - MONTECCHIO MAGGIORE

FRIULI VENEZIA GIULIA

Gorizia
• E.CO. ELETTRONICA - VIA F.LLI COSSAR 23
Trieste
• AVANZO GIACOMO - P.ZZA CAVANA 7
• COMPUTER SHOP - VIA P. RETI 6
• COMPUTIGI - VIA XX SETTEMBRE 51
• CTI - VIA PASCOLI 4
Udine
• MOFERT 2 - VIA LEOPARDI 21
• R.T. SISTEM UDINE - VIA L. DA VINCI 99
Provincia di Udine
• IL PUNTO ELETTRONICO - VIA VENDRAMIN 184 - LATISANA
• IDRENO MATTIUSSI &C. - VIA LICINIANA 58 - MARTIGNACCO

TRENTINO ALTO ADIGE

Bolzano
• COMPUTER POINT - VIA ROMA 82/A
• MATTEUCCI PRESTIGE - VIA MUSEO 54
Provincia di Bolzano
• RADIO MAIR-ELECTRO - VIA CENTRALE 70 - BRUNICO
• ELECTRO RADIO HENDRICH - VIA DELLE CORSE 106 - MERANO
• ERICH KONTSCHIEDER - PORTICI 313 - MERANO
• ELECTRO TAPPEINER - P.ZZA PRINCIPALE 90 - SILANDRO
Trento
• CRONST - VIA G. GALILEI 25
Provincia di Trento

• AL RISPARMIO - C.SO VERONA 138 - ROVERETO

LIGURIA
Genova
• ABM COMPUTER - P.ZZA DE FERRARI 24 rosso
• CAPRIOTTI G. - IA MAMIANI 4r - SAMPIERDARENA
• C.tro ELET. - VIA CHIARAVAGNA 10 R - VIA SESTRI 69R
• COM.le SOTTORIPA - VIA SOTTORIPA 115/117
• FOTOMONDIAL - VIA DEL CAMPO 3-5-9-11-13 r
• LA NASCENTE - VIA SAN LUCA 4/1
• PLAY TIME - VIA GRAMSCI 3/5/7 rosso
• RAPPR-EL - VIA BORGORATTI 23 R
Imperia
• CASTELLINO - VIA BELGRANO 44
Provincia di Imperia
• CENTRO HI-FI VIDEO - VIA DELLA REPUBBLICA 38 -SANREMO
• CASTELLINO - VIA GENOVA 48 - VENTIMIGLIA
La Spezia
• I.L. ELETTRONICA - VIA V. VENETO 123
Provincia di La Spezia
• I.L. ELETTRONICA - VIA AURELIA 299 - FORNOLA DI VEZZANO
Savona
• CASTELLINO - C.SO TARDY E BENECH 101
Provincia di Savona
• CELESIA ENZA - VIA GARIBALDI 144 - LOANO

EMILIA
Bologna
• EUROELETTRICA - VIA RANZANI 13/2
• MINNELLA ALTA FEDELTA' - VIA MAZZINI 146/2
• MORINI & FEDERICI - VIA MARCONI 28/C
• STERLINO - VIA MURRI 73/75
Provincia di Bologna
• S.C. COMPUTERS - VIA E. FERMI 4 - CASTEL SAN PIETRO
• S.P.E. INFORMATICA - VIA DI MEZZO PONENTE 385 - CREVALCORE
• ARCHIMEDE SISTEMI - VIA EMILIA 124 - S. LAZZARO DI SAVENA
Modena
• CO - EL - VIA CESARI 7
• ORSA MAGGIORE - P.ZZA MATTEOTTI 20
• VIDEO VAL WILLY COMPUTERS - VIA CANALETTO 223
Provincia di Modena
• NEW MEDIA SYSTEM - VIA ROMA 281 - SOLIERA
Parma
• BABARELLI G. - VIA B. PARENTE 14/A/B
Provincia di Parma
• PONGOLINI - VIA CAVOUR 32 - FIDENZA
Piacenza
• COMPUTER LINE - VIA G. CARDUCCI 4
• DELTA COMPUTER - VIA M. DELLA RESISTENZA 15/G

TEGGIO EMILIA
• COMPUTERLINE - VIA SAN ROCCO 10/C
• POOL SHOP - VIA EMILIA S. STEFANO 9/C
Provincia di Reggio Emilia
• MACCHIONI - VIA STATALE 467 - CASALGRANDE
ROMAGNA
Ferrara
• BUSINESS POINT - VIA CARLO MAYER 85
Forlì
• COMPUTER VIDEO CENTER - VIA CAMPO DI MARTE 122
Provincia di Forlì
• TOP BIT - VIA VENETO 12 - FORLIMPOPOLI
• COMPUTER HOUSE - V.LE TRIPOLI 193/D - RIMINI
• EASY COMPUTER - VIA LAGOMAGGIO 50 - RIMINI

REPUBBLICA S. MARINO
Ravenna
• COMPUTER HOUSE - VIA TRIESTE 134
Provincia di Ravenna
• ARGNANI - P.ZZA DELLA LIBERTA' 5/A - FAENZA
• ELECTRON INFORMATICA - VIA F.LLI CORTESI 17 - LUGO
• P.L.Z. INFORMATICA - P.ZZA SERCOGNANI 6 - FAENZA

TOSCANA
Arezzo
• DELTA SYSTEM - VIA PIAVE 13
Firenze
• ATEMA - VIA BENEDETTO MARCELLO 1a-1b
• ELETTRONICA CENTOSTELLE - VIA CENTO STELLE 5/a-b
• HELP COMPUTER - VIA DEGLI ARTISTI 15-A
• TELEINFORMATICA TOSCANA -VIA BRONZINO 36
Provincia di Firenze
• WAR GAMES - VIA R. SANZIO 126/A - EMPOLI
• NEW EVM COMPUTER - VIA DEGLI INNOCENTI 2 - FIGLINE VALDARNO
• C.tro INFOR. - VIA ZNOJMO 41 - PONTASSIEVE
• COSCI F.LLI - VIA ROMA 26 - PRATO
• BARBAGLI C. ELET. - VIA F. BONI 80 - PRATO
Grosseto
• COMPUTER SERVICE - VIA DELL'UNIONE 7
Livorno
• ETA BETA - VIA SAN FRANCESCO 30
• FUTURA 2 - VIA CAMBINI 19
Provincia di Livorno
• PUNTO ROSSO - VIA BARONTINI 28 - PIOMBINO
Provincia di Lucca
• IL COMPUTER - V.LE COLOMBO 216 - LIDO DI CAMAIORE
• SANTI VITTORIO - VIA ROMA 23 - S. ROMANO GARFAGNANA
• TOP GAMES - VIA S. ANDREA 122 - VIAREGGIO
Massa
• EURO COMPUTER - P.ZZA G. BERTAGNINI 4
Carrara
• RADIO LUCONI - VIA ROMA 24/B
Pisa
• ELECTRONIC SERVICE - VIA DELLA VECCHIA TRANVIA 10
• PUCCINI S.- CP 1199 (RAG.SOC. MAREX) - VIA C.CAMMEO 64
• TONY HI-FI - VIA CARDUCCI
Provincia di Pisa
• M.C. INFORMATICA - VIA DEL CHIESINO 4 - PONTEDERA (PI)
Pistoia
• ELECTRONIC SHOP - VIA DEGLI SCALZI 3
Provincia di Pistoia
• ZANNI &C. - C.SO ROMA 45 - MONTECATINI T.

Siena
• R. BROGI - P.ZZA GRAMSCI 28
• VIDEO MOVIE - VIA GARIBALDI 17
Provincia di Siena
• ELETTRONICA di BIFOLCHI - VIA DI GRACCIANO NEL CORSO 111 - MONTEPULCIANO

LAZIO
• CENTRO INF. - D.R.R. srl - TEL. 06-5565672

UMBRIA
Perugia
• MIGLIORATI - VIA S. ERCOLANO 3-10
Provincia di Perugia
• COMPUTER STUDIO'S - VIA IV NOVEMBRE 18/A - BASTIA UMBRA
• WARE - VIA DEI CASCERI 31 - CITTA'DI CASTELLO
Terni
• CGS SOFTWARE HOUSE - VIA DONIZETTI 71/A

BASILICATA
Matera
• G. GAUDIANO ELECTRONICS - VIA ROMA ang. XX SETTEMBRE 1

PUGLIA
Bari
• ARTEL - VIA GUIDO D'ORSO 9
• COMPUTER'S ARTS - V.LE MEUCCI 12/B
• PAULICELLI S. & F. - VIA FANELLI 231/C
Provincia di Bari
• F. FAGGELLA - C.SO GARIBALDI 15 - BARLETTA
• G FAGGELLA - P.ZZA D'ARAGONA 62A - BARLETTA
• LONUZZO G. - VIA NIZZA 21 - CASTELLANA
• TECNOUFF. - VIA RICASOLI 54 - MONOPOLI
• TANGORRA N. - C.SO V.EMANUELE 130/B - TRIGGIANO
Brindisi
• MARANGI E NICCOLI - VIA PROV. SAN VITO 165
Provincia di Brindisi
• MILONE G. - VIA S.F. D'ASSISI 219 - FRANCAVILLA FONTANA
Foggia
• BOTTICELLI G. - VIA SAV POLLICE 2
• E.C.I. COMPUTER - VIA ISONZO 28
• LA TORRE - V.LE MICHELANGELO 185
Provincia di Foggia
• IL DISCOBOLO - VIA T. SOLIS 15 - SAN SEVERO
Lecce
• BIT - VIA 95 REGG.NTO FANTERIA 87/89
Provincia di Lecce
• TECNO UFFICIO - P.ZZA GIOVANNI XXIII 10 - GALLIPOLI
• CEDOK INFORMATICA - VIA UMBERTO I 116 - TRICASE
Taranto
• ELETTROJOLLY C.tro - VIA DE CESARE 13
• TEA - TEC. ELET. AV. - VIA R. ELENA 101

CAMPANIA
Provincia di Avellino
• FLIP FLOP - VIA APPIA 68 - ATRIPALDA
Benevento
• E.CO. INF. - VIA PEPICELLI 21/25
Caserta
• ENTRY POINT - VIA COLOMBO 31
• O.P.C. - VIA G. M. BOSCO 24
Provincia di Caserta
• M.P. COMPUTER - VIA NAPOLI 30 - MADDALONI
• DAMIANO - C.SO V. EMANUELE 23 -ORTA DI ATELLA
• FUSCO B. - VIA NAPOLI 24 - VAIRANO PATERNORA (FRAZ. VAIRANO SCALO)
• LINEA CONTABILE - VIA OSPEDALE 72/76 - SESSA A. (CE)
Napoli
• BABY TOYS - VIA CISTERNA DELL'OLIO 5/BIS
• CASA MUSICALE RUGGIERO - P.ZZA GARIBALDI 74 (INT. STAZ. F.F. S.S.)
• C.tro ELET. CAMPANO - VIA EPOMEO 121
• CI.AN - GALLERIA VANVITELLI 32
• CINE NAPOLI - VIA S. LUCIA 93/95
• DARVIN - CALATA SAN MARCO 26
• GIANCAR 2 - P.ZZA GARIBALDI 37
• ODORINO - L.GO LALA 22 A-B
• R 2 - VIA F. CILEA 285
• SAGMAR - VIA S. LUCIA 140
• TOP VIDEO - TOP COMPUTER - VIA S. ANNA DEI LOMBARDI 12
• VIDEOFOTOMARKET - VIA S. BRIGIDA 19
Provincia di Napoli
• ELECTRONIC DAY - VIA DELLE PUGLIE 17 - CASORIA
• TUFANO - S.S. SANNITICA 87 KM 7 - CASORIA
• SOF SUD - V.LE EUROPA 59 - CASTEL/MARE DI STABIA
• ELETTRONICA 2000 - C.SO DURANTE 40 - FRATTAMAGGIORE
• SPADARO - VIA ROMANI 93 - MADONNA DELL'ARCO
• GATEWAY - VIA NAPOLI 68 - MUGNANO
• VISPINI & DI VUOLO - VIA A.ROSSI 4 - POMPEI
• SPY CASH & CARRY - P.ZZA ARENELLA 6/A - NAPOLI
• NUOVA INFORMATICA SHOP - VIA LIBERTA' 185/191 - PORTICI
• BASIC COMPUTER - C.SO GARIBALDI 34 - POZZUOLI
• V.C. - C.SO SECONDIGLIANO 562/B - SECONDIGLIANO
• F. ELETTRONICA - VIA SARNO 102 - STRIANO
• TECNO - VIA V. VENETO 48 - TORRE DEL GRECO
Salerno
• COMPUMARKET - VIA BELVEDERE 35
• COMPUTER MARKET - C.SO VITTORIO EMANUELE 23
Provincia di Salerno
• KING COMPUTER - VIA OLEVANO 56 - BATTIPAGLIA
• DIMER POINT - V.LE AMENDOLA 36 - EBOLI
• IACUZIO F. - VIA MUNICIPIO 14 - MERCATO SAN SEVERINO
• COMPUTER SERVICE - VIA L.DA VINCI 81 - SCAFATI

CALABRIA
Catanzaro
• C. & G. COMPUTER - VIA F. ACRI 28
• PAONE S. & F. - VIA F. ACRI 93/99
Provincia di Catanzaro
• COMPUTER HOUSE - VIA BOLOGNA (L.GO OSPEDALE) - CROTONE
• RIOLO F.LLI - VIA VENEZIA 1/7 - CROTONE
• ING. FUSTO S. - C.SO NICOTERA 99 - LAMEZIA TERME
Cosenza
• MAISON DE L'INFORMATIQUE - VIA PASQUALE ROSSI 34/C
• SIRANGELO COMP. - VIA N. PARISIO 25
Provincia di Cosenza
• HI-FI ALFANO G. - VIA BALDACCHINI 109 - AMANTIA
• ELIGIO ANNICCHIARICO &C. - VIA ROMA 21 - CASTROVILLARI
• ALFA COMPUTER - VIA NAZIONALE 341/A - CORIGLIANO SCALO
REGGIO CALABRIA
• CONTROL SYSTEM - VIA S.F DA PAOLA 49 D
• SYSTEM HOU. - VIA FIUME ang. PALESTINO 1
Provincia di Reggio Calabria
• COMPUTER SHOP - V.LE MATTEOTTI 36/38 - LOCRI
• PICIEFFE - C.SO F. S. ALESSIO 19 - TAURIANOVA
SICILIA
• CENTRO INF. - ITALSOFT SRL - TEL. 0935-696090

DIMENSIONE
AVVENTURA

JONATHAN

OGNI MESE
IN EDICOLA